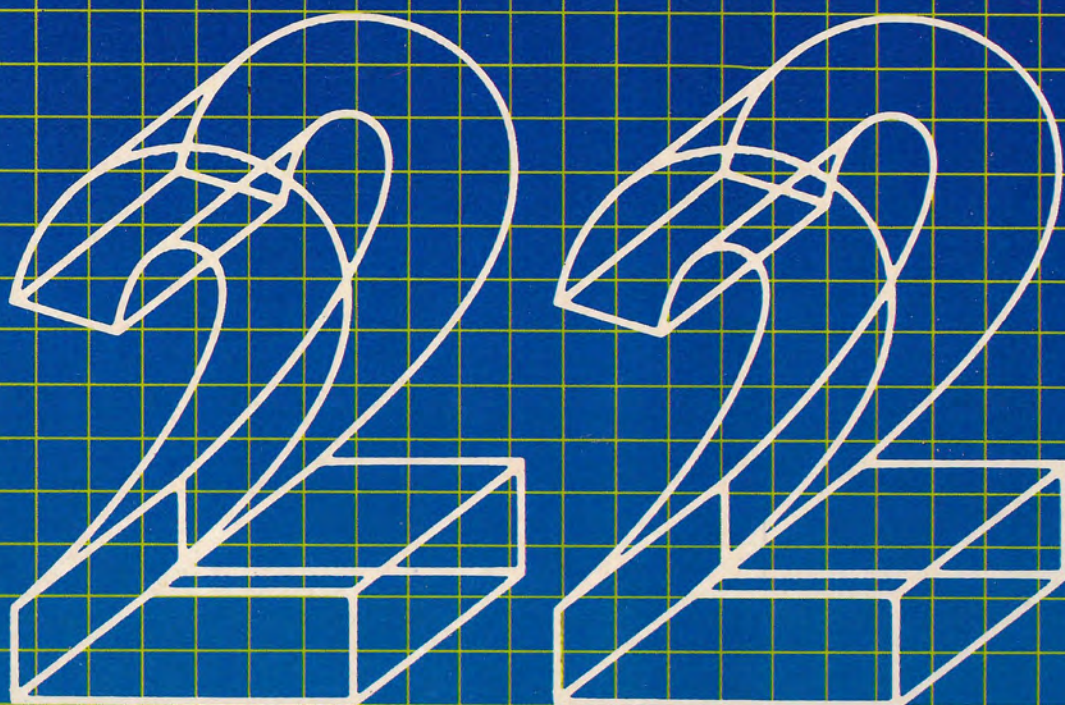


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



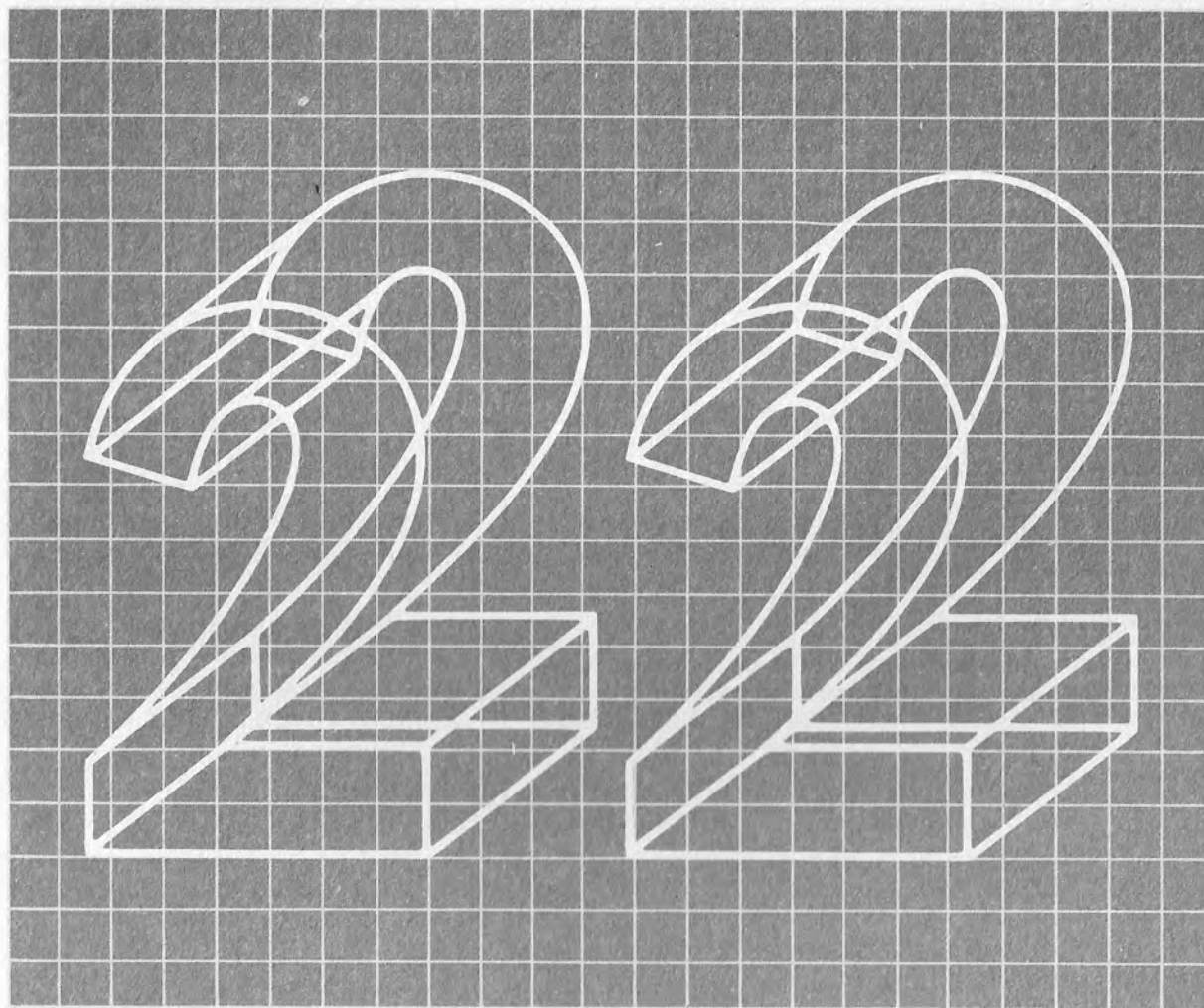
TRUCOS DE PROGRAMACION
Y DE DISEÑO DE GRAFICOS

EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 22. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: José María Esteban, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-125-8

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

MATEU CROMO. PINTO (Madrid).

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Junio, 1987

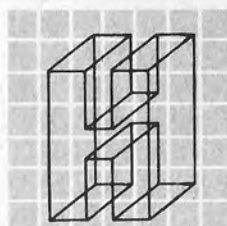
P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	17
■ TRUCOS Y RUTINAS BASICAS	25
■ EL TALLER DEL HARDWARE	35
■ APRENDER CON EL ORDENADOR	43
■ PEQUEÑA HISTORIA DE LA INFORMATICA	49
■ TEMAS MONOGRAFICOS DE VANGUARDIA	52
■ TERMINOLOGIA	56
■ VOCABULARIO DE INFORMATICA	58

EXPERIENCIA Y PRACTICAS EN LOGO

■ La recursividad a fondo



ASTA ahora el tipo de recursividad que hemos visto es la que denominamos recursividad de cola (la llamada recursiva se encuentra al final del procedimiento).

Que la llamada se encuentre al final del procedimiento no es nada obligatorio, detrás de ella podemos incluir las órdenes que queramos.

La cuestión es:

¿Cuándo se ejecutan las órdenes que hay detrás de la llamada recursiva?

Empecemos con un ejemplo sencillo:

```
? PARA DEMO :L
> ESCRIBE :L
> SI :L = 3 [ ALTO ]
> DEMO :L + 1
> ESCRIBE :L
> FIN
```

Si ejecutamos el procedimiento anterior con la llamada:

```
? DEMO 1
```

en la pantalla aparece lo siguiente:

```
1
2
3
2
1
```

Primero se cuenta hasta 3 y luego se descuenta hasta 1.

Antes de empezar con el estudio de este procedimiento recordemos cómo actúa la orden ALTO.

ALTO tiene la ejecución del procedimiento en curso y devuelve el control al procedimiento que efectuó la llamada (procedimiento anterior), prosiguiendo en éste la ejecución a partir del punto en el que se produjo la ruptura de secuencia.

Veamos ya detenidamente el esquema del desarrollo sustituyendo las variables por el valor que tienen.

```
PARA DEMO 1
ESCRIBE 1
SI 1 = 3 [ALTO]
DEMO 2
ESCRIBE 1
FIN
  PARA DEMO 2
  ESCRIBE 2
  SI 2 = 3 [ALTO]
  DEMO 3
  ESCRIBE 2
  FIN
    PARA DEMO 3
    ESCRIBE 3
    SI 3 = 3 [ALTO]
    DEMO 4
    ESCRIBE 3
    FIN
```

Cuando ejecutamos, por primera vez, el procedimiento, la variable L toma el valor 1; por tanto, en la pantalla aparece un 1; como L no es igual a 3, se pasa a ejecutar la línea siguiente, que es la llamada recursiva, el valor que pasamos al procedimiento llamado es L + 1, que en este caso es 2.

Este conjunto de acciones se repetirán hasta que L valga 3.

Cuando L es igual a 3, la condición se cumple y se ejecuta la orden ALTO, que, como hemos dicho, devuelve el control al procedimiento anterior, que seguirá ejecutándose a partir de la línea de llamada. Por tanto, se ejecutarán las dos órdenes que hasta ahora no se

Se pueden incluir órdenes después de las llamadas recursivas.

EXPERIENCIA Y PRACTICAS EN LOGO

habían efectuado (ten en cuenta que el valor de la variable L en este procedimiento era y es 2).

A su vez la última orden del procedimiento (FIN) devuelve el control al procedimiento anterior, en el que también se ejecutan las órdenes pendientes.

Las órdenes que se encuentran detrás de la llamada recursiva únicamente no se ejecutan en el último procedimiento DEMO.

Incluyendo una entrada más al procedimiento DEMO, conseguiremos saltar la barrera del 3.

```
? PARA DEMO :L :M
> ESCRIBE :L
> SI :L = M [ ALTO ]
> DEMO :L + 1
> ESCRIBE :L
> FIN
```

L para indicar el límite inferior.

M para indicar el límite superior.

Ejecuta ahora el procedimiento DEMO de forma que cuente y descuenta más de tres valores, por ejemplo:

```
? DEMO 5 50
```

Veamos qué tal dibujamos con lo que hemos aprendido:

```
? PARA D2 :N :F
> SI :N = 0 [ SL PONRUMBO 0 PONX 1
  PONY -10 BL ALTO ]
> FIGURA :F
> D2 :N - 1 :F
> FIGURA :F
> FIN
```

Las entradas de D2 son:

:N número de figuras que componen el dibujo.

:F número de lados de cada figura.

Este procedimiento realiza:

1. N llamadas al procedimiento FIGURA.
2. Cuando (N = 0) varía la posición de la Tortuga.
3. N llamadas al procedimiento FIGURA.

Formando así un dibujo con $2 * N$ figuras.

El procedimiento FIGURA, como su nombre indica, se encarga de dibujar un figura de F lados.

```
? PARA FIGURA :F
> REPITE :F [ AV 30 GD 360 / :F ]
> SL
> AV 30
> GD 30
> BL
> FIN
```

Si queremos obtener el dibujo que realiza D2 con 12 figuras de 3 lados cada una te-
cleamos:

```
? BP D2 12 3
```

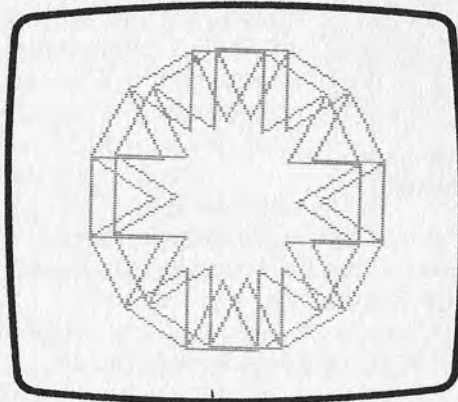


Fig. 1

0 uno con 12 figuras de cinco lados cada una:

```
? BP D2 12 5
```

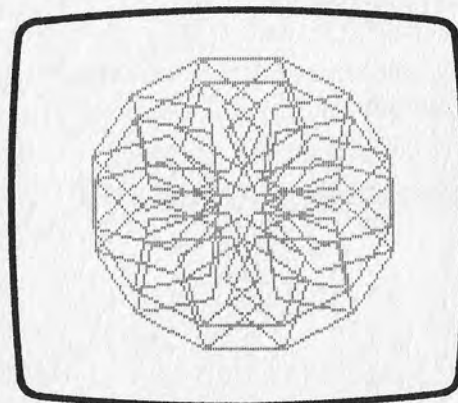


Fig. 2

También podemos definir un procedimiento para realizar, uno tras otro, distintos dibujos.

El Logo nos permite utilizar la impresora.

```

? PARA DIBUJOS
> HAZ "C 3
> REPITE 7 [ BP D2 12 :C HAZ "C :C +
  1 ESPERA 20 ]
> FIN

```

Inicializamos la variable que determina los lados de la figura 3, y repetimos siete veces la ejecución de D2, incrementando cada vez la variable C en 1, consiguiendo así que la figura que compone el dibujo sea cada vez distinta.

Un ejemplo típico de recursividad es la creación de árboles.

El siguiente procedimiento dibuja una de las ramas del árbol que más tarde dibujaremos.

```

? PARA AR :L :R
> SI :R = 0 [ ALTO ]
> AV :L
> GI 45
> AR :L / 2 :R - 1
> GD 45
> RE :L
> FIN

```

Ejecutamos el procedimiento:

```

? AR 50 3

```

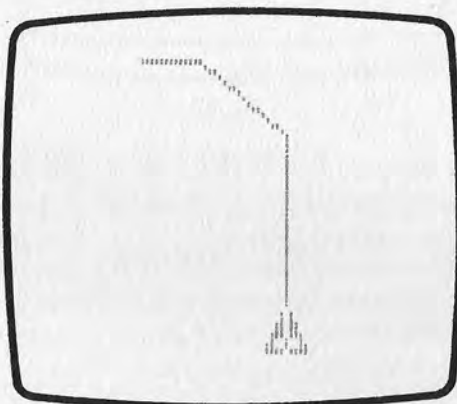


Fig. 3

El procedimiento AR tiene dos entradas:

L longitud de la rama.

R nivel de desarrollo del árbol.

Dependiendo del valor que le asigne-

mos a la variable de entrada R, el árbol que dibujemos tendrá más o menos ramificaciones.

La primera vez que se ejecuta el procedimiento, L tiene el valor 50 y R el valor 3, se está ejecutando AR 50 3.

— La condición no se cumple $R = 0$, la orden de los corchetes no se ejecuta.

— La Tortuga avanza L pasos y gira 45 grados a la izquierda. En la pantalla aparece:

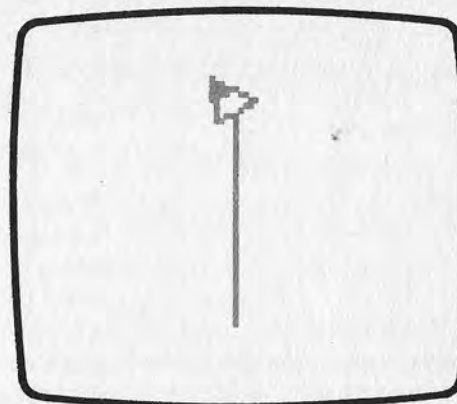


Fig. 4

— Llamada recursiva.

AR (:L / 2) (:R - 1), que sustituyendo por los valores es AR 25 2

Estas acciones las determinan las cuatro primeras órdenes del procedimiento, y se repetirán hasta que la variable R sea igual a 0.

En el momento en que $R = 0$ se ejecuta la orden ALTO, que devuelve el control al procedimiento anterior.

Acciones que se realizan:

— La Tortuga gira 45 grados a la derecha y retrocede L pasos. El valor de la variable L es el que tenía antes de que se ejecutase la llamada recursiva.

— FIN. Esta orden retorna el control al procedimiento anterior, en el que se ejecutarán las órdenes que aparecen detrás de la llamada recursiva.

Y así sucesivamente hasta que se llega al procedimiento de grado superior, es decir, al primero que se ejecutó (AR 50 3).

Estas últimas órdenes consiguen devol-

Si deseas imprimir algo, primero tienes que activar el canal de comunicación con la impresora.

EXPERIENCIA Y PRACTICAS EN LOGO

ver la Tortuga, pasando por donde había pasado, a su posición inicial.

El árbol completo se realiza incluyendo en el procedimiento anterior dos órdenes más:

1. Un giro a la derecha de 90 grados.
2. Una llamada recursiva: ARBOL :L / 2 :R - 1.

Incluyendo estas órdenes conseguimos dibujar otra rama junto a cada una de las que antes habíamos dibujado.

```
? PARA ARBOL :L :R
> SI :R = 0 [ ALTO ]
> AV :L
> GI 45
> ARBOL :L / 2 :R - 1
> GD 90
> ARBOL :L / 2 :R - 1
> GI 45
> RE :L
> FIN
```

Si lo ejecutamos con los mismos valores del procedimiento anterior, veremos claramente la diferencia:

```
? ARBOL 50 3
```

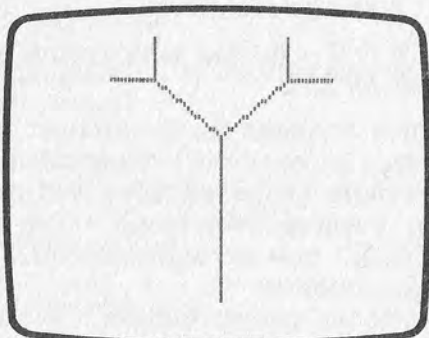


Fig. 5

Un árbol con más ramificaciones:

```
? ARBOL 50 6
```

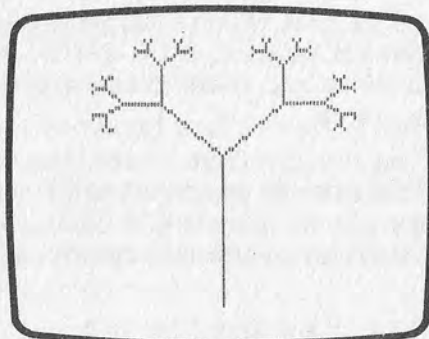


Fig. 6

Estableciendo otro tipo de variación en las longitudes de las ramas, los árboles que dibujemos tendrán formas diferentes a la anterior.

Por ejemplo, en el siguiente procedimiento la longitud de una rama será la longitud de la anterior menos dos puntos:

```
? PARA ARBOL :L :R
> SI :R = 0 [ ALTO ]
> AV :L
> GI 45
> ARBOL :L - 2 :R - 1
> GD 90
> ARBOL :L - 2 :R - 1
> GI 45
> RE :L
> FIN
```

```
? ARBOL 20 4
```

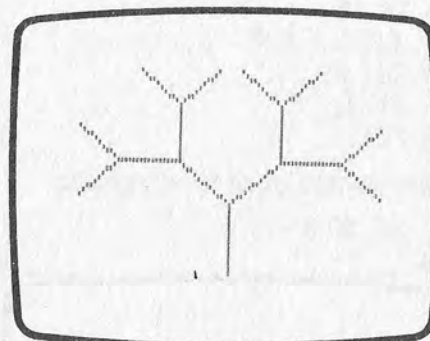


Fig. 7

Orientemos la Tortuga hacia distintos puntos antes de dibujar un árbol y conseguiremos curiosos dibujos.

El procedimiento DIBUJO se encarga de ello, solamente tenemos que ejecutarlo indicando tres datos de entrada:

N número de árboles que queremos dibujar.

L longitud de la rama.

R nivel de desarrollo del árbol.

```
? PARA DIBUJO :N :L :R
> BP
> HAZ "A 360 / :N
> HAZ "C 0
> REPITE :N [ PONRUMBO :C ARBOL
: L :R HAZ "C :C + :A ]
> FIN
```

```
? DIBUJO 4 20 3
```

Podemos obtener listados de procedimientos.

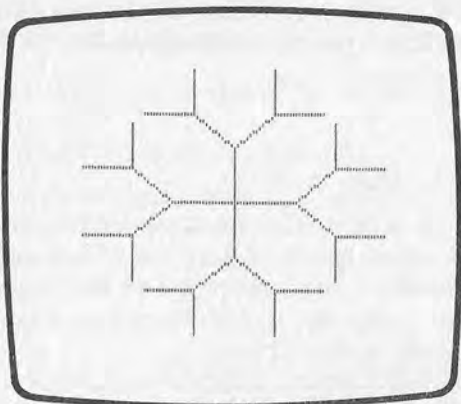


Fig. 8

? DIBUJO 8 25 4

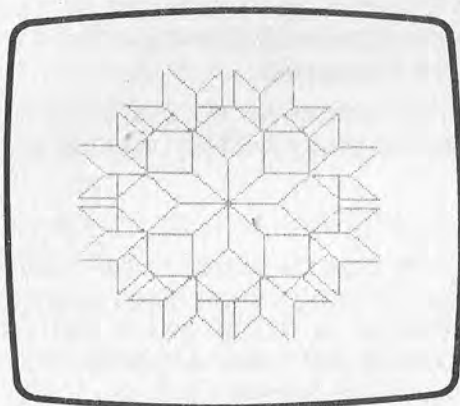


Fig. 9

■ El Logo y la impresora

Al igual que la mayoría de los lenguajes de programación, el Logo nos permite utilizar la impresora para obtener mediante ella una copia en papel, de procedimientos, lista de variables, etc.

Para realizar esto lo primero que hay que tener es una impresora adecuada y compatible con el ordenador con el que se está trabajando.

Mediante un cable, la impresora se conecta al ordenador, estableciendo de esa forma una comunicación entre ambos. Es conveniente conectar la impresora al ordenador antes de empezar una sesión de trabajo, cuando el ordenador y la impresora se encuentran desconectados de la corriente eléctrica.

Una vez que la conexión está realizada, hay que controlar el canal de acceso a la impresora. El control consiste en activar o desactivar este canal para utilizar o no la impresora.

La orden que controla el canal de la impresora es:

? IMPRESORA n

n es un número que dependiendo de su valor, indicará si se desea activar el canal o, por el contrario, desactivarlo.

Para activarlo introducimos la orden:

? IMPRESORA 1

y para desactivarlo:

? IMPRESORA 0

Una vez que conocemos la forma de indicar que se desea o no trabajar con la impresora, conozcamos cómo y qué podemos obtener por medio de ella.

En el momento que el canal de la impresora es activado, todo lo que se escribe a continuación o se obtiene en la pantalla se imprime.

Si queremos sacar por la impresora el listado de un determinado fichero que contenga procedimiento, previamente tenemos que salvar (guardar), en un soporte de información externo, el fichero a imprimir.

Una vez guardado el fichero, lo imprimimos mediante la orden:

? IMARCHIVO "nombre de fichero.

IMARCHIVO es la abreviatura de IMPrime ARCHIVO.

El nombre de fichero (o archivo) ha de ser el mismo que hemos dado al fichero en el cual hemos guardado los procedimientos.

Todos los procedimientos que contiene el fichero se imprimirán.

La razón de que el fichero tenga que estar guardado en un soporte de información, es que al realizar la impresión se está leyendo desde el soporte dicho fichero y no de la memoria del ordenador. Por ello, la unidad lectora deberá estar activada.

Si lo que queremos es obtener el listado de un procedimiento definido, introducimos la orden:

? IM "nombre de procedimiento

No todas las impresoras sirven para todos los ordenadores.

EXPERIENCIA Y PRACTICAS EN LOGO

IM es la abreviatura de IMprime.

Esta orden presenta en la pantalla el listado del procedimiento que se especifica. Si la impresora está activada, lo obtenemos también por la impresora.

Si deseamos obtener el listado de más de un procedimiento:

? IM [nombre proc.1 nombre proc.2 ...]

Supongamos que definimos el siguiente procedimiento:

? PARA EJEMPLO
> BP
> OT
> BL
> REPITE 4 [AV 10 GI 90]
> FIN

Si queremos obtener una copia en papel de él, tenemos que realizar los siguientes pasos:

1.º Activar el canal de la impresora:

? IMPRESORA 1

2.º Ordenar la impresión del fichero:

? IM "CUADRADO

El listado del procedimiento aparece en pantalla y en la impresora.

El listado impreso será:

? IM "EJEMPLO

PARA EJEMPLO

BP

OT

BL

REPITE 4 [AV 10 GI 90]

FIN

Si a continuación no queremos obtener nada más por la impresora, desactivamos el canal:

? IMPRESORA 0

Todo lo que hemos visto es perfectamente válido si posees un ordenador MSX.

Si posees un PC-Compatible, los cambios que hay que realizar son:

Para activar el canal de la impresora:

? ECO "LPT1

Algunas versiones de Logo para estos ordenadores permiten también dar la orden:

? ECO "LPT2

o bien

? ECO "prn

La orden ECO es la encargada de activar un canal (para el caso de la impresora) o bien de abrir un fichero en el que deseamos guardar todas las instrucciones que se escriban desde ese momento.

Si el nombre que se especifica a continuación de ECO no es uno de los que hemos visto, por ejemplo:

? ECO "CASA

Estamos indicando que se abra un fichero llamado CASA, en el que se va a guardar una copia de todos los caracteres que aparecen en la pantalla.

Para desactivar el canal de la impresora o bien cerrar un fichero, introducimos la orden:

? NOECO

A la hora de obtener una copia impresa de un fichero que contenga procedimientos utilizamos la misma orden que para los MSX, pero en este caso debemos especificar la extensión de fichero:

? IMARCHIVO "CUADRADO.LOG

Si dispones de un SPECTRUM; utilizamos:

? CONIMPRESORA

Activa el canal de la impresora y ordena que se imprima todo lo que sigue a continuación de ser introducida la orden.

Para desactivar el canal y dejar de imprimir.

? SINIMPRESORA

■ Cuadro resumen

— IMPRESORA n (Para los MSX)

Activa o desactiva, dependiendo del valor que demos a n, el canal de comunicación con la impresora.

Para obtener el listado de un fichero, tiene que estar guardada en un soporte de información.

n=1, activa el canal.
n=0, desactiva el canal.

- ECO "puerto (para los PC-Compatibles)-
ECO "nombre de fichero

Activa el canal de la impresora si se especifica como puerto, LPT1.

Abre un fichero con el nombre que se especifica, al cual van a ser enviados los caracteres que aparecen en la pantalla.

- NOECO (para los PC-Compatibles)

Desactiva el canal de la impresora si anteriormente fue activado, o bien cierra el fichero abierto dejando de enviar caracteres.

- COINMPRESORA (para el SPECTRUM)

Activa el canal de comunicación con la impresora y todo lo que aparezca a continuación será impreso.

- SINIMPRESORA (para el SPECTRUM)

Desactiva el canal de comunicación con la impresora.

- IMARCHIVO "nombre de fichero

Se obtiene una copia en papel del contenido del fichero que se especifica.

- IM "nombre de procedimiento
IM [nombre proc.1 nombre proc.2 ...]

Se obtiene en la pantalla un listado de los procedimientos que se especifican. Si la impresora está activada, se obtiene también un listado en papel.

■ Ejercicios

1. Definir un procedimiento recursivo que escriba en la pantalla los 10 primeros números naturales seguidos de los 10 primeros números negativos.

Números naturales = números positivos

Cuando lo consigas añade una entrada al procedimiento para poder indicar, como

dato al ejecutar el procedimiento, el número de números que se quieren escribir.

Nota: No se tiene en cuenta el 0.

2. Utilizando la recursividad dibuja un cuadrado de un color y encima de él otro de otro color.
3. Intenta realizar el siguiente dibujo:

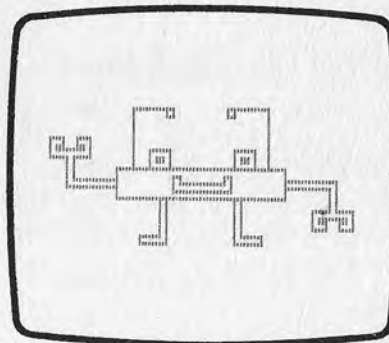


Fig. 10

4. Se pretende, una vez dibujadas una serie de bombillas, simular su encendido rellenándolas. Una vez que la última bombilla se ha encendido, empezarán a apagarse en el orden inverso a como se encendieron.
5. ¿Son correctas las órdenes siguientes?

? PARA N1

```
> MT
> REPITE 4 [ AV 30 GD 90 ]
> N1
> REPITE 36 [ AV 3 GD 10 ]
> FIN
```

? PARA N2

```
> BP
> MT
> ESCRIBE :N2
> N2
> FIN
```

? PARA N3 :N

```
> SI :N < 1 [ ALTO ]
> AV :N
> GI 60
> N3 :N / 2
> GD 60
> RE :N
> FIN
```

? PARA N4

```
> N4
```

Conecta la impresora cuando ésta y el ordenador estén apagados.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> REPITE 10 [ AV 10 N4 ]
> FIN

? PARA N5 :N :M :Ñ
> SI :N = 5 [ N5 :N :M :Ñ ]
> SI :M = 5 [ N5 :N :M :Ñ ]
> SI :Ñ = 5 [ N5 :N :M ]
> FIN
```

■ Solución a los ejercicios

1:

Primero definimos un procedimiento de inicialización, desde el que llamamos al procedimiento que escribe los números.

```
? PARA NUM
> BT
> HAZ "N 1
> NUMEROS :N
> FIN
```

El procedimiento NUMEROS tiene un parámetro de entrada:

```
N primer número
? PARA NUMEROS :N
> SI :N > 10 [ ALTO ]
> ESCRIBE :N
> NUMEROS :N + 1
> ESCRIBE -(11 - :N)
> FIN
```

Obtén los 20 números tecleando:

```
? NUM
```

Incluimos una entrada a los dos procedimientos.

M número de números a escribir.

```
? PARA NUM :M
> BT
> HAZ "N1
> NUMEROS :N :M
> FIN
```

Sustituimos en NUMEROS del valor 10 por la variable M:

```
? PARA NUMEROS :N :M
> SI :N > :M [ ALTO ]
> ESCRIBE :N
> NUMEROS :N + 1 :M
```

```
> ESCRIBE -((:M + 1) - :N)
> FIN
```

Para obtener cinco números positivos y cinco negativos tecleamos:

```
? NUM 5
```

Para obtener 100

```
? NUM 100
```

2:

Definimos un procedimiento para borrar la pantalla, inicializar el color del lápiz e inicializar la variable N:

```
? PARA C :L
> BP
> PONCL 1
> HAZ "N 1
> CUADRADO :N :L
> FIN
```

Desde este procedimiento llamamos al procedimiento CUADRADO, indicando junto a N la longitud del lado del cuadrado.

```
? PARA CUADRADO :N :L
> AV :L
> GD 90
> SI :N = 4 [ PONCL 2 AV :L ALTO ]
> CUADRADO :N + 1 :L
> GD 90
> AV :L
> FIN
```

3:

Para realizar el dibujo vamos a dibujar un procedimiento que dibuje cada una de las partes de que está compuesto.

En primer lugar, el cuerpo:

```
? PARA CUERPO
> SL
> PONX -25
> BL
> REPITE 2 [ AV 10 GD 90 AV 50 GD 90 ]
> FIN
```

Para dibujar los ojos:

```
? PARA OJOS
> SL
> PONPOS [ -15 10 ]
```

Una llamada recursiva puede hacerse desde cualquier parte del procedimiento.

```

> REPITE 2 [ REPITE 4 [ BL AV 6 GD
  90 ] SL PONX
> SL
> PONPOS [ -13 12 ]
> REPITE 2 [ REPITE 4 [ BL AV 2 GD
  90 ] SL PONX 12 ]
> FIN

```

Las antenas las dibuja el siguiente procedimiento:

```

? PARA ANTENAS
> SL
> PONPOS [ -20 -10 ]
> BL
> AV 20 GD 90 AV 10
> REPITE 4 [ AV 2 GD 90 ]
> SL
> PONPOS [ 20 10 ]
> PONRUMBO 0
> BL
> AV 20 GI 90 AV 10
> REPITE 4 [ AV 2 GI 90 ]
> FIN

```

Para dibujar la boca:

```

? PARA BOCA
> SL
> PONPOS [ -8 -3 ]
> PONRUMBO 0
> BL
> AV 4 GD 90
> AV 2 GD 90
> AV 2 GI 90
> AV 13 GI 90
> AV 2 GD 90
> AV 2 GD 90
> AV 4 GD 90
> AV 17
> FIN

```

El procedimiento PATAS dibuja las piernas del marciano:

```

? PARA PATAS
> SL
> PONPOS [ -10 0 ]
> PONRUMBO 180
> BL
> AV 15 GD 90
> AV 8 GD 90
> AV 2 GD 90
> AV 6 GI 90
> AV 13

```

```

> SL
> PONPOS [ 10 0 ]
> PONRUMBO 180
> BL
> AV 15 GI 90
> AV 8 GI 90
> AV 2 GI 90
> AV 6 GD 90
> AV 13
> FIN

```

Por último, definimos los procedimientos que van a dibujar los brazos. Definimos dos: BRAZOS, que sitúa a la Tortuga en la posición y con el rumbo adecuado, para dibujar cada brazo, y BRAZO, que es el que lo dibuja:

```

? PARA BRAZOS
> SL
> PONPOS [ -25 4 ]
> PONRUMBO 270
> BRAZO
> SL
> PONPOS [ 25 6 ]
> PONRUMBO 90
> BRAZO
> FIN

```

```

? PARA BRAZO
> BL
> AV 15 GD 90
> AV 10 GI 90
> AV 5 GD 90
> AV 6 GD 90
> AV 4 GD 90
> AV 2 GD 90
> AV 2 GI 90
> AV 2 GI 90
> AV 8 GI 90
> AV 2 GI 90
> AV 2 GD 90
> AV 2 GD 90
> AV 4 GD 90
> AV 6 GD 90
> AV 5 GI 90
> AV 8 GI 90
> AV 13
> FIN

```

Una vez que tenemos definidos todos, definimos un último para agruparlos:

```

? PARA MARCIANO
> BP

```

Durante el tiempo que dura la impresión, no se puede trabajar con el ordenador.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> OT
> CUERPO
> OJOS
> ANTENAS
> BOCA
> PATAS
> BRAZOS
> FIN
```

Para ejecutarlo:

```
? MARCIANO
```

4:

En primer lugar, vamos a definir un procedimiento para dibujar una bombilla:

```
? PARA BOMBILLA
> SL
> PONRUMBO 0
> PONY 0
> PONCL 2
> BL
> REPITE 2 [ AV 1 GD 90 AV 2 GD 90 ]
> AV 1 GI 90
> AV 3 GD 90
> REPITE 4 [ AV 8 GD 90 ]
> SL
> AV 8 GI 90
> RE 5
> BL
> REPITE 20 [ AV 3 GD 18 ]
> SL
> PONX COORX + 20
> FIN
```

Si lo ejecutamos obtenemos una bombilla como ésta:

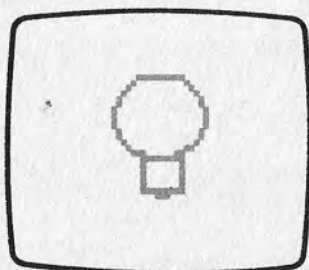


Fig. 11

El siguiente procedimiento es el que se encarga de dibujar centradas las bombillas que nosotros digamos. El número de bombillas a dibujar será el que indique :N

```
? PARA BOMBILLAS :N
> BP
```

```
> SL
> HAZ "A (-:N * 18) / 2
> PONX :A
> REPITE :N [ BOMBILLA ]
> INTERRUPTOR
> FIN
```

Este procedimiento hace una llamada a INTERRUPTOR, que es el que va a simular el encendido y el apagado de las bombillas. Para ello situamos a la Tortuga en el interior de la primera bombilla y la ordenamos que la rellene de un color determinado. Este mismo proceso lo realiza con las restantes bombillas.

Una vez que están todas encendidas (rellenas), realizamos el mismo proceso, pero a la inversa, y con el color del fondo de la pantalla:

```
? PARA INTERRUPTOR
> SL
> PONX :A
> PONY 15
> PONRUMBO 90
> PONCL 1
> REPITE :N [ BL RELLENA SL AV 22 ]
> PONCL FONDO
> REPITE :N [ SL RE 22 BL RELLENA ]
> FIN
```

Si no posees un PC-Compatible, cambia la orden:

```
> PONCL FONDO
```

por

```
> PONCL 3
```

Si queremos que aparezcan seis bombillas, introducimos:

```
? BOMBILLAS 6
```

En la pantalla obtenemos:

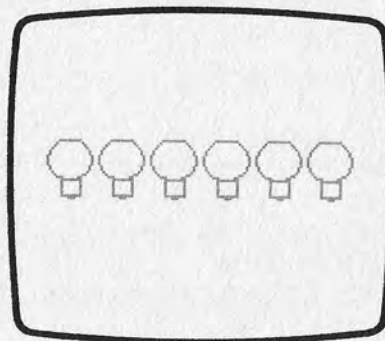


Fig. 12

El canal de comunicación recibe también el nombre de puerto.

5:

```
— ? PARA N1
> MT
> REPITE 4 [ AV 30 GD 90 ]
> N1
> REPITE 36 [ AV 3 GD 10 ]
> FIN
```

INCORRECTO.

Aunque el LOGO no mostrará ningún mensaje de error, no podemos considerar que el procedimiento sea correcto.

La orden que hay detrás de la llamada recursiva no se ejecutará nunca, pues no hay ninguna orden condicional que dé paso a su ejecución.

```
— ? PARA N2
> BP
> MT
> ESCRIBE :N2
> N2
> FIN
```

INCORRECTO.

El LOGO devuelve un mensaje de error:

N2 NO TIENE VALOR EN N2

Intentamos escribir el contenido de N2 sin haberla inicializado.

```
— ? PARA N3 :N
```

```
> SI :N < 1 [ ALTO ]
> AV :N
> GI 60
> N3 :N / 2
> GD 60
> RE :N
> FIN
```

CORRECTO.

La Tortuga recorrerá un cierto camino y luego retrocederá por él hasta llegar a la posición inicial.

```
— ? PARA N4
> REPITE 10 [ AV 10 N4 ]
> FIN
```

CORRECTO.

La Tortuga avanzaría cada vez 10 pasos y no se detendría jamás.

```
— ? PARA N5 :N :M :Ñ
> SI :N = 5 [ N5 :N :M :Ñ ]
> SI :M = 5 [ N5 :N :M :Ñ ]
> SI :Ñ = 5 [ N5 :N :M ]
> FIN
```

INCORRECTO.

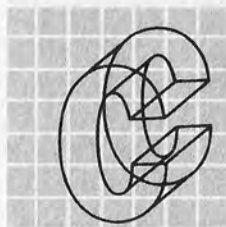
La acción a realizar en el caso de que se cumpla la última condición, hace una llamada a procedimiento N5, pero falta indicar el valor de la variable :Ñ.

■ **Desactiva la impresora cuando termines de trabajar con ella.**

■ **Los listados de impresora son de gran ayuda.**

■ **Utiliza la recursividad para el diseño de arcos.**

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



CONTINUANDO con lo que vimos en el tomo anterior, vamos a ver a continuación cómo utilizar el color en el AMSTRAD. Este ordenador se caracteriza por sus tres modos distintos de pantalla y por el distinto número de

colores que podemos representar en cada uno de ellos.

■ El uso del color en el Amstrad

El AMSTRAD es el ordenador personal que más colores tiene del mercado. Con el AMSTRAD podemos utilizar hasta 27 colores distintos. Gracias a esto, se puede decir que podemos dibujar cualquier cosa que se nos antoje. Los colores, como todo en el ordenador, están numerados de forma que si queremos decirle al ordenador que queremos utilizar un color, sólo tengamos que dar el número de dicho color. Esto es mucho más sencillo, y más lógico, que decir su nombre. Los colores están agrupados empezando por el más oscuro (negro) y terminando por el más claro (blanco brillante). En un monitor de fósforo verde (sin colores) los colores se verán como tonos de verde que se van haciendo más claros según aumentamos el número de color.

Los colores de que dispone el AMSTRAD son los siguientes:

- Color 0 = Negro
- Color 1 = Azul oscuro
- Color 2 = Azul oscuro brillante
- Color 3 = Rojo
- Color 4 = Magenta o morado

- Color 5 = Malva o lila
- Color 6 = Rojo brillante
- Color 7 = Púrpura
- Color 8 = Magenta o morado brillante
- Color 9 = Verde
- Color 10 = Cian o azul claro
- Color 11 = Azul cielo
- Color 12 = Amarillo
- Color 13 = Blanco
- Color 14 = Azul pastel
- Color 15 = Naranja
- Color 16 = Rosa
- Color 17 = Magenta o morado pastel
- Color 18 = Verde brillante
- Color 19 = Verde marino
- Color 20 = Cian o azul claro brillante
- Color 21 = Verde lima
- Color 22 = Verde pastel
- Color 23 = Cian o azul claro pastel
- Color 24 = Amarillo brillante
- Color 25 = Amarillo pastel
- Color 26 = Blanco brillante

El único problema que tenemos es que estos 27 colores no se pueden emplear a la vez. Tenemos que elegir unos cuantos que serán los que podremos utilizar. El número de colores que podemos utilizar depende del tipo de pantalla.

MODO 0 = 20 columnas por línea = 16 colores

MODO 1 = 40 columnas por línea = 4 colores

MODO 2 = 80 columnas por línea = 2 colores

Esto es debido a que el ordenador utiliza la misma cantidad de memoria para almacenar los distintos tipos de pantalla. Como los

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

colores ocupan memoria, al tener siempre la misma, el número de colores disminuye con el aumento de la resolución.

A continuación vamos a ver un programa

que nos muestra los distintos colores de que dispone el AMSTRAD. El programa nos cambia el color del borde de la pantalla y nos imprime un mensaje diciéndonos qué color es.

```
100 REM *****
110 REM * PROGRAMA DEMO PARA *
120 REM * VER LOS 27 COLORES *
130 REM * DEL AMSTRAD.      *
140 REM *****
150 REM
160 MODE 2
170 CLS
180 PRINT "LEYENDO DATAS."
190 PRINT
200 PRINT "ESPERE UN MOMENTO."
210 RESTORE
220 DIM A$(27)
230 FOR I=1 TO 27
240   READ A$(I)
250 NEXT I
260 REM
270 REM *** NOMBRES DE LOS COLORES ***
280 REM
290 DATA "Negro","Azul oscuro","Azul oscuro brillante"
300 DATA "Rojo","Magenta o Morado","Malva o Lila"
310 DATA "Rojo brillante","Purpura","Magenta o Morado brillante"
320 DATA "Verde","Cian o Azul claro","Azul cielo"
330 DATA "Amarillo","Blanco","Azul pastel"
340 DATA "Naranja","Rosa","Magenta o Morado pastel"
350 DATA "Verde brillante","Verde marino"
360 DATA "Cian o Azul claro brillante","Verde lima"
370 DATA "Verde pastel","Cian o Azul claro pastel"
380 DATA "Amarillo brillante","Amarillo pastel"
390 DATA "Blanco brillante"
400 REM
410 REM *** PROGRAMA DEMOSTRACION ***
420 REM
430 CLS
440 FOR I=0 TO 26
450   CLS
460   PRINT "ESTE COLOR ES: ";A$(I+1)
470   BORDER I
480   PRINT
490   PRINT "PULSA UNA TECLA PARA CONTINUAR"
500   LET B$=INKEY$
510   IF B$="" THEN GOTO 500
520 NEXT I
530 CLS
540 BORDER 0
550 PRINT "QUIERES VERLO DE NUEVO? (S/N)"
560 LET B$=INKEY$
570 IF B$="" THEN GOTO 560
580 IF B$="S" OR B$="s" THEN GOTO 430
590 IF B$<>"n" AND B$<>"B" THEN GOTO 560
600 CLS
610 FOR I=1 TO 10
620   FOR J=0 TO 26
630     BORDER J
640     FOR K=1 TO 100
650       NEXT K
660     NEXT J
670 NEXT I
680 BORDER 0
690 END
```

Una vez vistos los colores vamos a ver cómo utilizarlos. Para empezar tenemos que imaginarnos que la pantalla del ordenador fuese un papel. Dicho papel no ocupa toda la pantalla de nuestro monitor, sino que sólo ocupa la parte central. Alrededor de ésta se encuentra una especie de anillo cuadrado que llamaremos BORDE. En este BORDE no podemos imprimir ni dibujar, pero, como se vio en el programa 1, sí podemos cambiarlo de color.

Lo mismo ocurre con el resto de la pantalla (la hoja de papel); también la podemos cambiar de color. Si nos seguimos imaginando que la pantalla es un papel, como es lógico, podremos escribir en él y la tinta también podrá tener color.

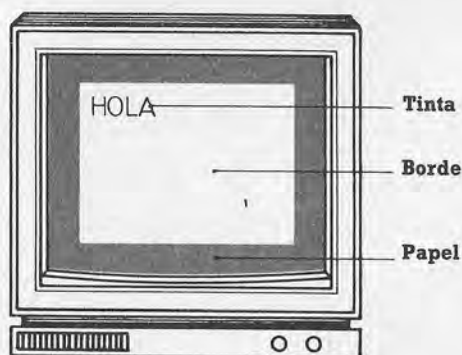


Fig. 1.

Si esto se entiende es muy fácil entender el resto. Para cambiar el color del borde, como ya hemos visto en el programa 1, lo único que hay que hacer es utilizar el comando:

BORDER n

donde 'n' es un número entero entre 0 y 26 y que le dice al ordenador de qué color queremos poner el borde. El efecto es inmediato. Según se ejecuta esta instrucción, el color del borde cambia al color elegido.

Hemos dicho un poco más arriba que no se podían utilizar nunca los 27 colores simultáneamente, y que dependía del tipo de pantalla el número de colores simultáneos que podríamos utilizar. Como es lógico, no podemos hacer que el borde tenga más de un color a la vez, pero sí podemos asignarle cualquiera de los 27 colores de que disponemos, estemos en el modo de pantalla que estemos.

Referente al color de la tinta y al del papel, el asunto se complica un poco. Antes de cambiar cualquiera de ellos, tenemos que decirle al ordenador qué colores vamos a utilizar.

Por ejemplo, si nos encontramos en el modo de pantalla número uno, como tenemos cuatro colores, tenemos que decirle al AMSTRAD qué cuatro colores vamos a utilizar. Esto lo hacemos gracias a la sentencia INK. Con ella haremos corresponder a cada uno de los cuatro colores de que disponemos el color que queremos.

Los cuatro colores de que disponemos están numerados del 0 al 3. Así, si queremos que el color 0 sea amarillo, como el color amarillo, en la tabla general de colores, tiene el número 12, pondremos lo siguiente:

INK 0,12

con esto hacemos que el color 0 sea amarillo. Esto puede parecer un poco complicado. Lo que hay que intentar es no confundir los números de la tabla general de colores con los números de los colores que podemos utilizar.

Una vez definido el color 0 como amarillo, para utilizarlo tendremos que decirle al ordenador que queremos utilizar el color 0 y no el 12. Este último no lo entendería.

Como ejemplo vamos a definir los cuatro colores de que disponemos. Introduce la siguiente orden directa:

INK 0,0:INK 1,2:INK 2,9:INK 3,13

con ello hemos definido el color 0 como amarillo, el 1 como azul oscuro brillante, el 2 como verde y el 3 como blanco. Al hacer esto, algunas cosas, en la pantalla, habrán cambiado. No hay por qué preocuparse. Vamos a ver cómo utilizar estos cuatro colores.

Para cambiar el color del PAPEL (la hoja de papel), tenemos el siguiente comando:

PAPER n

donde 'n' es un número entre 0 y el número máximo de colores que nos permita el tipo de pantalla en el que nos encontramos. En el modo 0 podrá tomar cualquier valor entre 0 y 16. En el modo 1 entre 0 y 3 y en el modo 2 entre 0 y 1.

Para cambiar el color de la tinta con la que escribiremos utilizaremos el comando:

PEN n

donde 'n' puede tomar cualquier valor entre 0 y el número máximo de colores que nos permita la pantalla en la que nos encontramos.

Si en cualquiera de estos dos comandos ponemos un número mayor del permitido, el ordenador dividirá dicho número entre el máximo de colores permitidos y pondrá el co-

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

lor del resto que nos quede. Por ejemplo, si en el modo 1 de pantalla ponemos:

PAPER 4

el ordenador cambiará el color del papel a color 0. Esto es así porque 4 dividido entre 4 es igual a 1 y nos da de resto 0. Este será el color

que tomará el papel. Si en vez de eso ponemos:

PAPER 7

el color del papel será 3 ($7/3=1$ y de resto 3).

A continuación vamos a ver otro programa que nos muestra cómo utilizar estos dos nuevos comandos.

```
100 REM *****
110 REM * PROGRAMA DEMO PARA VER *
120 REM * VER EL USO DE PEN Y *
130 REM * PAPER EN EL AMSTRAD. *
140 REM *****
150 REM
160 MODE 0
170 CLS
180 PRINT "LEYENDO DATAS."
190 PRINT
200 PRINT "ESPERE UN MOMENTO."
210 RESTORE
220 DIM A$(27)
230 FOR I=1 TO 27
240   READ A$(I)
250 NEXT I
260 REM
270 REM *** NOMBRES DE LOS COLORES ***
280 REM
290 DATA "Negro","Azul oscuro","Azul oscuro brillante"
300 DATA "Rojo","Magenta o Morado","Malva o Lila"
310 DATA "Rojo brillante","Purpura","Magenta o Morado brillante"
320 DATA "Verde","Cian o Azul claro","Azul cielo"
330 DATA "Amarillo","Blanco","Azul pastel"
340 DATA "Naranja","Rosa","Magenta o Morado pastel"
350 DATA "Verde brillante","Verde marino"
360 DATA "Cian o Azul claro brillante","Verde lima"
370 DATA "Verde pastel","Cian o Azul claro pastel"
380 DATA "Amarillo brillante","Amarillo pastel"
390 DATA "Blanco brillante"
400 REM
410 REM *** PROGRAMA DEMOSTRACION ***
420 REM
430 LET C1=0
440 LET C2=15
450 FOR I=C1 TO C2
460   INK I,I
470 NEXT I
480 FOR I=C1 TO C2
490   FOR J=C1 TO C2
500     PEN J
510     PAPER I
520     PRINT "PAPEL = ";A$(I+1);" TINTA = ";A$(J+1)
530     FOR K=1 TO 100
540       NEXT K
550     NEXT J
560 NEXT I
570 IF C1=0 THEN LET C1=16:LET C2=26:GOTO 450
580 MODE 1
590 FOR I=0 TO 3
600   INK I,I
610 NEXT I
620 END
```

Como se puede apreciar al observar la ejecución del programa, cada carácter puede tener dos colores. Uno el de la tinta (la forma del carácter) y otro el del papel (el fondo sobre el que se dibuja el carácter). Por ello, podemos decir que cada posición de carácter de la pantalla puede tener sus propios colores. Lo único que hay que hacer es cambiar el color de la tinta o del papel (o de los dos) antes de escribir algo, para que aparezca de los colores deseados.

El programa 3 nos muestra cómo cada carácter de una palabra puede tener un color distinto.

```

100 REM *****
110 REM * PROGRAMA DEMO PARA VER *
120 REM * QUE CADA POSICION DE *
130 REM * CARACTER EN EL AMSTRAD *
140 REM * PUEDE TENER DOS COLORES *
150 REM *****
160 REM
170 MODE 1
180 CLS
190 INPUT "QUE QUIERES QUE ESCRIBA? ";A$
200 MODE 0
210 INK 0,0
220 PAPER 0
230 CLS
240 FOR J=0 TO 15
250   PAPER J
260   LET CO=1

```

```

270   FOR I=1 TO LEN(A$)
280     PEN CO
290     PRINT MID$(A$,I,1);
300     LET CO=CO+1
310     IF CO=15 THEN LET CO=1
320   NEXT I
330   PRINT
340 NEXT J
350 END

```

Lo que hacemos en este programa es ir cambiando el color de cada letra del mensaje que introduce el usuario. Una vez que hemos impreso todo el mensaje, cambiamos el color del papel y lo volvemos a imprimir cambiando de nuevo el color de las letras. Este proceso se repite 16 veces hasta que se ha impreso el mensaje con los 16 colores distintos de papel. Habrá momentos en que algunas letras no se vean. Esto ocurrirá cuando la tinta y el papel sean iguales.

Gracias a la instrucción INK, el AMSTRAD nos permite cambiar el color de un mensaje que ya esté impreso. Si, por ejemplo, hemos impreso la palabra HOLA en amarillo sobre fondo negro (siendo el amarillo el color 1 y el negro el color 0), si queremos cambiar el color de dicha palabra, no tenemos que volver a escribirla, sólo necesitamos utilizar de nuevo la instrucción INK para que cambie de color. El programa 4 nos demuestra un ejemplo muy sencillo de esto.

```

100 REM *****
110 REM * PROGRAMA DEMO PARA CAMBIAR *
120 REM * LA TINTA DE UN MENSAJE SIN *
130 REM * TENER QUE VOLVER A ESCRIBIR *
140 REM * DICHO MENSAJE. *
150 REM *****
160 REM
170 MODE 1
180 INK 0,0
190 REM
200 INK 1,12:REM <-- Fijate en esta linea.
210 REM
220 INK 2,26
230 CLS
240 PEN 1
250 PAPER 0
260 PRINT "ESTE MENSAJE ESTA EN COLOR AMARILLO"
270 PRINT:PRINT:PRINT:PRINT:PRINT
280 PEN 2
290 PRINT "PULSA UNA TECLA PARA CAMBIAR EL COLOR DEL MENSAJE"
300 LET A$=INKEY$
310 IF A$="" THEN 280
320 REM
330 INK 1,9:REM <-- Fijate en esta linea.
340 REM

```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

350 PRINT
360 PRINT
370 PRINT "AHORA EL MENSAJE ES DE COLOR VERDE"
380 END

```

En este programa lo único que hacemos es que el color 1 sea igual al amarillo. Después imprimimos un mensaje y cambiamos el color 1 a verde. Como dicho mensaje está escrito en color 1, al cambiar su representación (no el valor, sino la forma en que vemos el color) también cambia el color.

Sólo nos queda por ver una cosa sobre el color. El parpadeo. El AMSTRAD nos permite que un mensaje que nosotros imprimamos sea parpadeante. Para ello no tenemos que imprimir el mensaje, borrarlo, volverlo a imprimir, volverlo a borrar y así sucesivamente. Podemos hacer que cualquier cosa que imprimamos o dibujemos en la pantalla esté parpadeando con sólo decirle al ordenador entre qué colores queremos que parpadee y la velocidad de dicho parpadeo.

Si queremos que el color del borde parpadee entre dos colores tendremos que utilizar la instrucción BORDER con el siguiente formato:

BORDER c1, c2

donde 'c1' es el primer color y 'c2' el segundo color. Si, por ejemplo, escribimos:

BORDER 12,0

el borde se pondrá de color amarillo, estará un rato con dicho color, se pondrá negro, estará otro rato en ese color, se volverá a poner amarillo y así sucesivamente y sin interrupción. El parpadeo no afecta al programa, por lo que no tenemos que estar pendientes de él pudiendo continuar de forma normal.

Si lo que queremos es que un determinado color de tinta sea parpadeante, entonces utilizaremos la instrucción INK. Esto quiere decir que si queremos imprimir un mensaje parpadeante entre dos colores, ambos colores forman un solo color. Aunque parezca un poco complicado, esto se entiende perfectamente con un ejemplo. Por ejemplo, si queremos imprimir la palabra HOLA de forma que su color parpadee entre el rojo y el cian, tendremos que hacer lo siguiente:

INK 1,3,10:PEN 1:PRINT "HOLA":PEN 2

con esta instrucción imprimimos la palabra HOLA en la pantalla de forma que su color vaya variando entre el rojo y el cian. Como

puede verse, lo que hemos hecho ha sido asignar el parpadeo entre estos dos colores al color 1 de la paleta. Por ello, todo lo que escribamos con color 1 será parpadeante entre el rojo y el cian.

La forma de utilizar el comando INK para definir colores parpadeantes es la siguiente:

INK c,c1,c2

donde 'c' es el color de la paleta que hemos elegido y que puede estar entre 0 y el número máximo de colores permitido en la pantalla en la que nos encontramos. 'c1' es el primer color de parpadeo de la tabla general de colores y 'c2' el segundo color de parpadeo también tomado de dicha tabla.

Un poco más arriba hemos indicado que la cadencia del parpadeo puede ser variada desde el programa. Para ello disponemos de otro comando del BASIC del AMSTRAD. Este es:

SPEED INK n1,n2

donde 'n1' es el tiempo que estará visible el color 'c1' y 'n2' es el tiempo que estará visible el color 'c2'. El tiempo se mide en centésimas de segundo, de manera que cada valor que pongamos vale por dos centésimas de segundo. Por ello, si escribimos:

SPEED INK 10,20

el primer color estará en la pantalla durante 20 centésimas de segundo (2×10), momento en el cual aparecerá en pantalla el segundo color, el cual permanecerá durante cuarenta centésimas de segundo (2×20). Una vez pasado este tiempo, volveremos a ver el primer color.

El programa 5 nos muestra en pantalla todo lo que hemos visto sobre el parpadeo. La pantalla parece volverse loca y los ojos se nos salen de las órbitas, pero el programa explica muy bien el funcionamiento del parpadeo.

```

100 REM *****
110 REM * PROGRAMA PARA VER EL USO *
120 REM * DEL PARPADEO. *
130 REM *****
140 REM

```

```

150 MODE 1
160 CLS
170 INK 0,0,1
180 INK 1,0,13
190 INK 2,13,1
200 INK 3,10,24
210 PAPER 0
220 PEN 1
230 CLS
240 PRINT
250 PRINT "AHORA ME VES (Y AL DE ABAJO NO)"
260 PRINT
270 PEN 2
280 PRINT "AL DE ARRIBA NO LE VES PERO A MI SI"
290 PRINT
300 PEN 3
310 PRINT "A MI ME VES SIEMPRE, PERO BAILANDO"
320 PRINT:PRINT:PRINT
330 PRINT "COMO ARREGLAS ESTO"
340 END

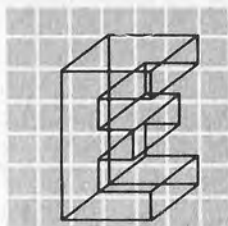
```

Para terminar os mostramos una tabla donde aparecen los colores que tenemos en cada modo y su correspondencia con los números de los colores de la tabla general cuando conectemos nuestro ordenador.

MODOS	COLOR	NÚMERO
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	23
24	24	24

Fig. 2. Referencias PAPEL/PLUMA/MODO/COLOR.

TRUCOS Y RUTINAS BASICAS



N el tomo anterior vimos un programa para definir SPRITES en el MSX. Estos SPRITES podían ser utilizados en modos de pantalla superiores al cero (0). A continuación vamos a ver un programa que nos permitirá definir nuestros propios caracteres en el MODO 1 de pantalla. En los demás modos de pantalla esto no es posible, porque la tabla que tiene almacenadas las formas de los caracteres se encuentra en la ROM. En este modo de pantalla el ordenador vuelca dicha información en la RAM. Por ello podemos modificarla a nuestro gusto para así tener otro juego de caracteres.

La utilidad de tener otro juego de caracteres es evidente. En muchos juegos vamos a necesitar dibujar cosas en pantalla de forma que no tenemos suficiente con los SPRITES de que disponemos. Por ejemplo, si queremos im-

primir una serie de paredes que tengan ladrillos, necesitaría muchos SPRITES que tuviesen todos la misma forma. Esto es porque un SPRITE no puede estar en más de una posición a la vez. Si definimos un carácter con forma de ladrillo, podremos imprimirlo todas las veces que nosotros queramos. Gracias a ello, podemos utilizar los SPRITES para definir aquellos personajes que han de moverse, y definir los caracteres para que actúen como decorado.

Con este programa podemos definir todos los caracteres que tiene el ordenador. Podemos realizar letras del espacio o letras góticas. Podemos variar la forma de los números para las puntuaciones. Gracias al programa podemos tener más de un juego de letras y de números. Como tenemos una serie de caracteres que muy raramente se utilizan (los que van desde el 127 hasta el 255), en éstos podemos definirnos nuestro segundo alfabeto y aún nos sobran caracteres para hacer lo que queramos con ellos.

```
1000 REM *****
1010 REM *
1020 REM *   E D I T O R   *
1030 REM *
1040 REM *       D E       *
1050 REM *
1060 REM * C A R A C T E R E S *
1070 REM *
1080 REM *       P A R A     *
1090 REM *
1100 REM *       M S X       *
1110 REM *
1120 REM *****
1130 REM
1140 REM *****
1150 REM *****
1160 REM * (c) ED. SIGLO CULTURAL *
1170 REM *****
1180 REM * (c) 1987 *
1190 REM *****
1200 REM *****
1210 REM
1220 SCREEN 1
1230 LOCATE 0,0,0
```

TRUCOS Y RUTINAS BASICAS

```
1240 COLOR 11,0,0
1250 CLEAR 300
1260 KEY OFF
1270 PRINT "      EDITOR DE CARACTERES"
1280 PRINT " ====="
1290 LOCATE 1,12
1300 PRINT "ESPERE UN MOMENTO POR FAVOR"
1310 DIM C(255,7)
1320 DI=BASE(7)
1330 FOR G=0 TO 255
1340   FOR I=0 TO 7
1350     C(G,I)=VPEEK(G*8+DI+I)
1360   NEXT I
1370 NEXT G
1380 REM
1390 REM *****
1400 REM * DEFINICION DE VARIABLES *
1410 REM *****
1420 REM
1430 MU=BA+173*8
1440 F=0
1450 FOR I=DI+171*8 TO DI+171*8+15
1460   READ A
1470   VPOKEI,A
1480 NEXT I
1490 DATA 0,126,66,66,66,66,126,0,0,126,126,126,126,126,126,0
1500 CLS
1510 LOCATE 15,3
1520 PRINT"CARACTER : "
1530 PU=65
1540 CX=0
1550 CY=0
1560 GOSUB 2450
1570 GOSUB 2150
1580 REM
1590 REM *****
1600 REM * MENU PRINCIPAL *
1610 REM *****
1620 REM
1630 LOCATE 0,9
1640 PRINT"Opciones:"
1650 PRINT
1660 PRINT"F1-Invierte.", "F6-Load.", "F2-Rota iz.", "F7-Save set.", "F3-Rota der.",
" F8-Save data.", "F4-Espejo hor.F9-Edita.", "F5-Espejo ver.F10-Borra.", STRING$(29,
"-")
1670 GOSUB 1950
1680 REM
1690 REM *****
1700 REM * FORMA DE CONTROL *
1710 REM *
1720 REM * 0=TECLADO
1730 REM * 1=JOYSTICK
1740 REM *****
1750 REM
1760 FC=0
1770 ON STRIG GOSUB 2360
1780 STRIG(FC) ON
1790 ON INTERVAL=25 GOSUB 2250
1800 ON KEY GOSUB 2920,2700,2810,3160,3030,3330,3560,3770,2550,4030
1810 FOR I=1 TO 10
1820   KEY(I) ON
1830 NEXT I
1840 M=STICK(FC)
1850 IF M<>0 THEN STRIG(FC) OFF:INTERVAL STOP:F=1:GOSUB 2250 ELSE 1840
```

```

1860 FOR CC=1 TO 40
1870 NEXT CC
1880 CY=(CY+(M=1)-(M=5) AND 7)
1890 CX=(CX-(M=3)+(M=7) AND 7)
1900 F=1
1910 GOSUB 2250
1920 INTERVAL ON
1930 STRIG(FC) ON
1940 GOTO 1840
1950 REM
1960 REM *****
1970 REM * PONE PARRILLA *
1980 REM *****
1990 REM
2000 INTERVAL OFF
2010 STRIG(FC) OFF
2020 LOCATE 0,0,0
2030 FOR G=0 TO 7
2040     T=128
2050     FOR N=0 TO 7
2060         IF (C(PU,G) AND T)=T THEN PRINTCHR$(172); ELSE PRINTCHR$(171);
2070         T=T/2
2080     NEXT N
2090     PRINT
2100 NEXT G
2110 INTERVAL ON
2120 STRIG(FC) ON
2130 BEEP
2140 RETURN
2150 REM
2160 REM *****
2170 REM * ASIGNA PARRILLA AL *
2180 REM * CARACTER 173. *
2190 REM *****
2200 REM
2210 FOR G=0 TO 7
2220     VPOKE MU+G,C(PU,G)
2230 NEXT G
2240 RETURN
2250 REM
2260 REM *****
2270 REM * INTERVAL *
2280 REM *****
2290 REM
2300 LOCATE CX,CY,0
2310 T$=""
2320 IF F THEN IF (C(PU,CY) AND 2^(7-CX))<>0 THEN T$=CHR$(172) ELSE T$=CHR$(171)
2330 PRINT T$
2340 F=-(F=0)
2350 RETURN
2360 REM
2370 REM *****
2380 REM * TRIGGER *
2390 REM *****
2400 REM
2410 VPOKE MU+CY,(VPEEK(MU+CY) XOR 2^(7-CX))
2420 C(PU,CY)=VPEEK(MU+CY)
2430 RETURN
2440 REM
2450 REM *****
2460 REM * PONE EL CARACTER *
2470 REM * ASIGNADO *
2480 REM *****
2490 REM

```

TRUCOS Y RUTINAS BASICAS

```
2500 INTERVAL OFF
2510 LOCATE 13,5
2520 PRINT PU;" ";TAB(18);"-";CHR$(PU);"-";CHR$(173)
2530 INTERVAL ON
2540 RETURN
2550 REM
2560 REM *****
2570 REM * EDITA EL CARACTER *
2580 REM * ELEGIDO. *
2590 REM *****
2600 REM
2610 GOSUB 4340
2620 LOCATE 0,17
2630 INPUT "Caracter a editar";A$
2640 PU=ASC(A$)
2650 GOSUB 2170
2660 GOSUB 2450
2670 GOSUB 1950
2680 GOSUB 4280
2690 RETURN
2700 REM
2710 REM *****
2720 REM * ROTACION IZQUIERDA *
2730 REM *****
2740 REM
2750 FOR I=0 TO 7
2760   C(PU,I)=(C(PU,I)*2 AND 254)+(C(PU,I) AND 128)/128
2770   VPOKE MU+I,C(PU,I)
2780 NEXT I
2790 GOSUB 1950
2800 RETURN
2810 REM
2820 REM *****
2830 REM * ROTACION DERECHA *
2840 REM *****
2850 REM
2860 FOR I=0 TO 7
2870   C(PU,I)=(C(PU,I)/2 AND 127)+128*(C(PU,I) AND 1)
2880   VPOKE MU+I,C(PU,I)
2890 NEXT I
2900 GOSUB 1950
2910 RETURN
2920 REM
2930 REM *****
2940 REM * INVERSION DEL CARACTER *
2950 REM *****
2960 REM
2970 FOR G=0 TO 7
2980   C(PU,G)=(C(PU,G) XOR 255)
2990   VPOKE MU+G,C(PU,G)
3000 NEXT G
3010 GOSUB 1950
3020 RETURN
3030 REM
3040 REM *****
3050 REM * ESPEJO VERTICAL *
3060 REM *****
3070 REM
3080 FOR G=0 TO 3
3090   T=C(PU,7-G)
3100   C(PU,7-G)=C(PU,G)
3110   C(PU,G)=T
3120 NEXT G
3130 GOSUB 1950
```

```

3140 GOSUB 2150
3150 RETURN
3160 REM
3170 REM *****
3180 REM * ESPEJO HORIZONTAL *
3190 REM *****
3200 REM
3210 FOR G=0 TO 7
3220 B$=BIN$(C(PU,G))
3230 B$=STRING$(8-LEN(B$),"0")+B$
3240   A$=""
3250   FOR I=1 TO 8
3260     A$=MID$(B$,I,1)+A$
3270   NEXT I
3280   C(PU,G)=VAL("&B"+A$)
3290   VPOKE MU+G,C(PU,G)
3300 NEXT G
3310 GOSUB 1950
3320 RETURN
3330 REM
3340 REM *****
3350 REM * LOAD CARACTERES *
3360 REM *****
3370 REM
3380 GOSUB 4340
3390 LOCATE 0,17
3400 INPUT "A partir del caracter ";C$
3410 INPUT "Nombre";N$
3420 T=ASC(C$)
3430 PRINT"Ponga en marcha el cassette.."
3440 OPEN "cas:"+N$ FOR INPUT AS #1
3450 IF EOF(1) THEN 3510
3460 FOR I=0 TO 7
3470   INPUT#1,C(T,I)
3480 NEXT I
3490 T=T+1
3500 GOTO 3450
3510 CLOSE#1
3520 GOSUB 4240
3530 GOSUB 1950
3540 GOSUB 2150
3550 RETURN
3560 REM
3570 REM *****
3580 REM * SAVE *
3590 REM *****
3600 REM
3610 GOSUB 4340
3620 LOCATE 0,17
3630 INPUT"A partir del caracter";C$
3640 T=ASC(C$)
3650 INPUT"Cuantos caracteres";N
3660 INPUT "Nombre";N$
3670 GOSUB 4140
3680 OPEN "cas:"+N$ FOR OUTPUT AS #1
3690 FOR G=T TO T+N-1
3700   FOR I=0 TO 7
3710     PRINT#1,C(G,I)
3720   NEXT I
3730 NEXT G
3740 CLOSE#1
3750 GOSUB 4240
3760 RETURN
3770 REM

```

TRUCOS Y RUTINAS BASICAS

```
3780 REM *****
3790 REM * SAVE COMO DATAS *
3800 REM *****
3810 REM
3820 GOSUB 4340
3830 LOCATE 0,17
3840 INPUT "A partir del caracter";N$
3850 T=ASC(N$)
3860 INPUT "Cuantos caracteres";N
3870 INPUT "Nombre";N$
3880 INPUT "Linea de comienzo";LI
3890 GOSUB 4140
3900 OPEN "cas:" + N$ FOR OUTPUT AS #1
3910 FOR G=T TO T+N-1
3920   A$=STR$(LI)+" DATA "
3930   FOR I=0 TO 6
3940     A$=A$+"&H"+HEX$(C(G,I))+", "
3950   NEXT I
3960   A$=A$+"&H"+HEX$(C(G,7))
3970   PRINT#1,A$
3980   LI=LI+10
3990 NEXT G
4000 CLOSE#1
4010 GOSUB 4240
4020 RETURN
4030 REM
4040 REM *****
4050 REM * BORRADO DE CARACTER *
4060 REM *****
4070 REM
4080 FOR G=0 TO 7
4090   C(PU,G)=0
4100   VPOKE MU+G,0
4110 NEXT G
4120 GOSUB 1950
4130 RETURN
4140 REM
4150 REM *****
4160 REM * MENSAJE DE CASSETTE *
4170 REM *****
4180 REM
4190 LOCATE 0,22
4200 PRINT "Ponga en marcha el cassette          y pulse una tecla";
4210 I$=INKEY$
4220 IF I$="" THEN 4210
4230 RETURN
4240 REM
4250 REM *****
4260 REM * BORRA LA VENTANA *
4270 REM * INFERIOR *
4280 REM *****
4290 REM
4300 LOCATE 29,23
4310 PRINT STRING$(203,127)
4320 INTERVAL ON
4330 RETURN
4340 REM
4350 REM *****
4360 REM * PREPARA INPUT *
4370 REM *****
4380 REM
4390 POKE &HF3F8,PEEK(&HF3FA)
4400 POKE &HF3F9,PEEK(&HF3FB)
4410 INTERVAL OFF
4420 RETURN
```

El funcionamiento del programa es muy sencillo y muy parecido al DEFINIDOR DE SPRITES. Este programa tiene algunas opciones más que el anterior, pero, en reglas generales, su utilización y la forma en que está hecho son las mismas.

Al ejecutar el programa nos aparecerá una pantalla organizada de la siguiente forma:

- En la esquina superior izquierda nos aparecerá una cuadrícula de 8×8 puntos que utilizaremos para definir nuestro carácter en tamaño gigante.

- En la esquina superior derecha nos aparecerá el carácter que estamos definiendo con su forma normal y con la nueva forma que le estamos dando.

- En la parte baja de la pantalla nos aparecerá el menú con las opciones de que disponemos.

Al igual que en el programa de SPRITES, para mover el cursor parpadeante que se encuentra dentro de la retícula donde tenemos que definir el carácter, utilizaremos las teclas del cursor. Para fijar un punto utilizaremos la barra espaciadora. Cuando la pulsemos, el punto sobre el que se encuentra el cursor se encenderá, si estaba apagado, o se apagará, si estaba encendido.

Según vayamos cambiando la forma del carácter, iremos viendo cómo va quedando a tamaño real con sólo mirar el carácter que está impreso en la esquina superior derecha de la pantalla.

Las opciones de las que disponemos con este programa están disponibles pulsando las teclas de función. Estas funciones, junto con las teclas que las realizan son:

F1—INVIERTE. Al pulsar esta tecla, todos los puntos que están encendidos se apagan y todos los que están apagados se encienden.

F2—ROTA A LA IZQUIERDA. Si pulsamos esta tecla de función el dibujo entero de la parrilla se moverá una posición hacia la izquierda. Si parte del dibujo desaparece por la izquierda, volverá a aparecer por la derecha.

F3—ROTA A LA DERECHA. Realiza la misma función que la tecla anterior, pero esta vez, en lugar de desplazarse el dibujo hacia la izquierda lo hará hacia la derecha. Si parte del dibujo se sale por la parte derecha, volverá a aparecer por la izquierda.

F4—ESPEJO HORIZONTAL. Se realiza un giro del carácter según un eje horizontal que se encuentra justamente en medio de dicho carácter. Con ello se consigue que lo que estaba arriba se ponga abajo y que lo que estaba abajo se ponga arriba. Gracias a este comando podemos tener letras en distintas direcciones. Podemos tener las letras boca abajo.

F5—ESPEJO VERTICAL. Realiza la misma operación que la tecla anterior, pero en este caso el eje de giro es vertical. Conseguimos así que lo que está a la derecha se coloque a la izquierda, y lo que está a la izquierda, a la derecha.

F6—Pulsando esta tecla podremos leer unos caracteres que previamente hemos definido y grabado. Para que el ordenador pueda leer los caracteres que hace tiempo habíamos definido, es necesario que hayamos grabado sus definiciones en formato interno y no como líneas DATA. Para grabar los caracteres en dicho formato tenemos que pulsar la tecla F7.

F7—Con esta tecla podemos grabar los caracteres que hemos definido de forma que más adelante y en otra ocasión podamos volver a leerlos con el programa para continuar con su definición o modificarlos. Si grabamos los caracteres con esta opción no podremos leerlos desde fuera de este programa. Si queremos grabar las definiciones de forma que las podamos utilizar en nuestros programas tendremos que pulsar la opción de F8.

F8—Nos permite grabar los caracteres que hemos definido en formato de líneas DATA. De esta forma, para utilizarlos sólo tenemos que realizar un MERGE de nuestro programa con el fichero que creamos al pulsar esta opción. Al pulsar esta tecla y también al pulsar F7, el ordenador nos preguntará desde qué carácter queremos grabar y cuántos. En este caso también nos preguntará qué número de líneas queremos que tenga la primera línea DATA que grabe el programa.

F9—Pulsando esta tecla de función, hacemos que un carácter que ya está definido nos aparezca en la parrilla para continuar su definición. El carácter que estábamos editando antes de pulsar esta tecla no se pierde, sino que queda almacenado automáticamente.

F10—Con esta opción borramos la pantalla y la dejamos limpia. También borramos el carácter que estamos definiendo.

Como se puede apreciar, no hay ninguna opción para almacenar el carácter que estamos definiendo en un carácter cualquiera.

TRUCOS Y RUTINAS BASICAS

Lo que sucede es que continuamente, cada vez que el usuario dibuja o borra un punto, se actualiza la forma de dicho valor. Por ello, pulsamos la opción que pulsemos, en cualquier momento el carácter que estábamos definiendo

no se pierde, pues se define al mismo tiempo que lo dibujamos.

A continuación, y para que el lector practique, aparecen los números del 0 al 9 redefinidos.

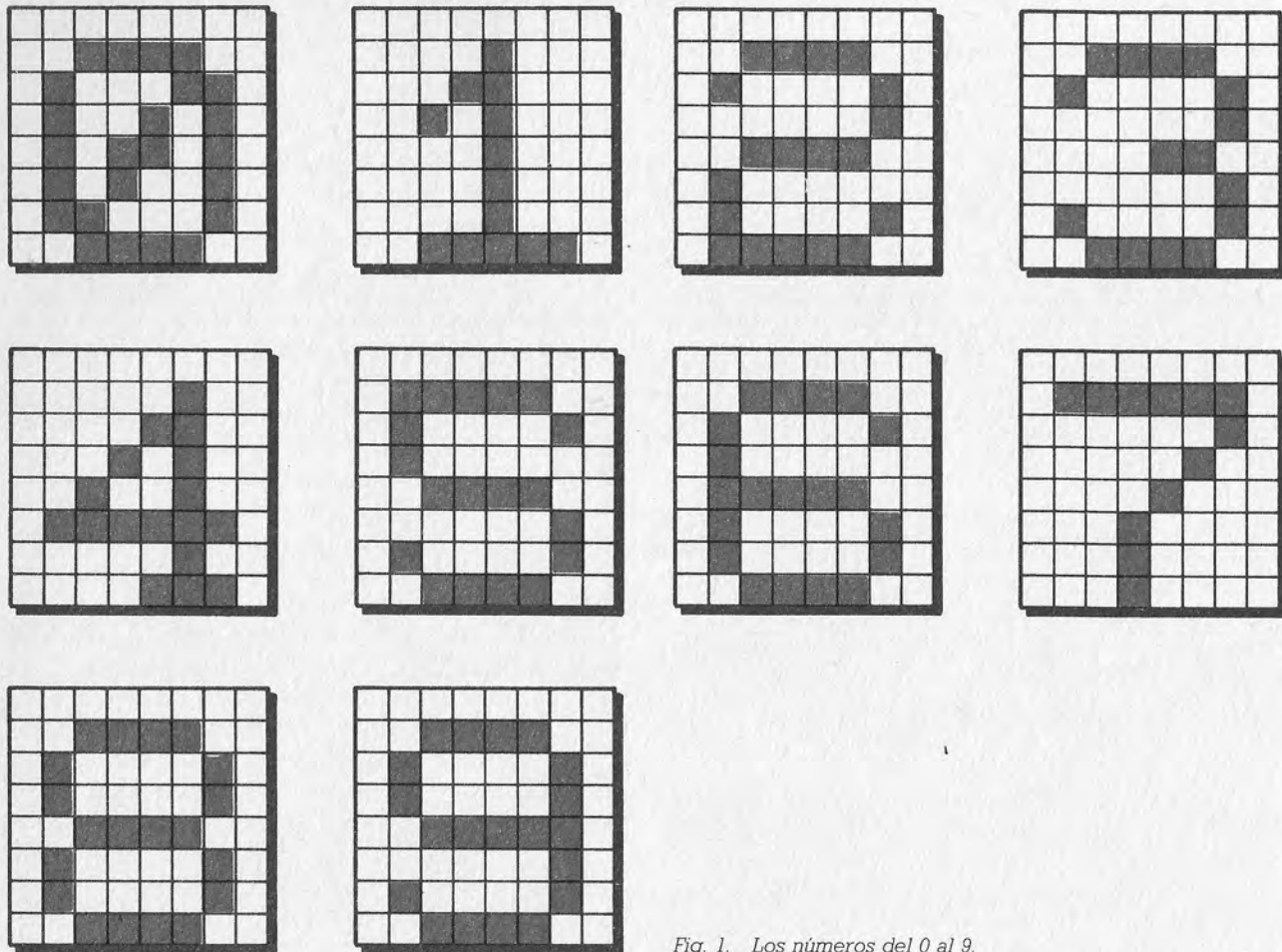


Fig. 1. Los números del 0 al 9.

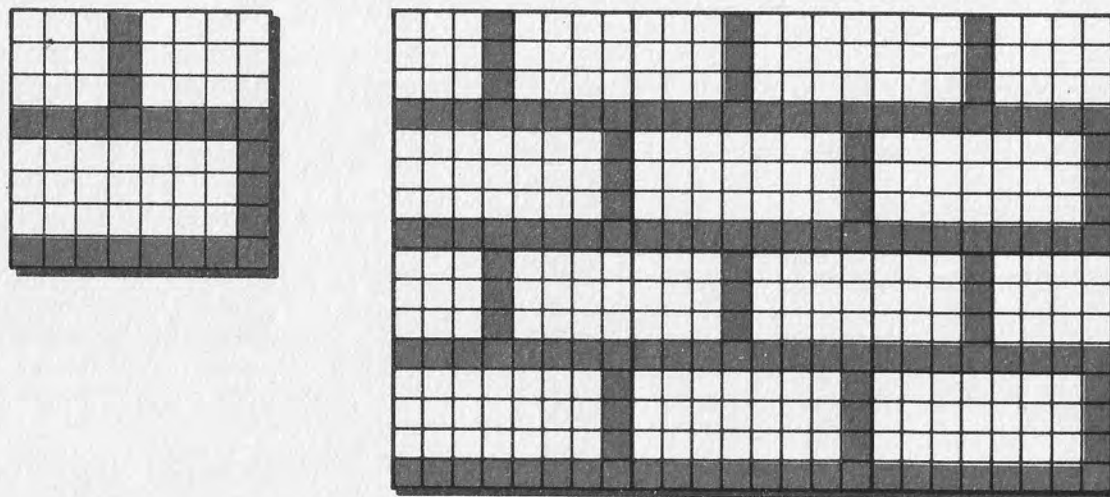


Fig. 2. Cómo definir un ladrillo y cómo quedaría una pared.

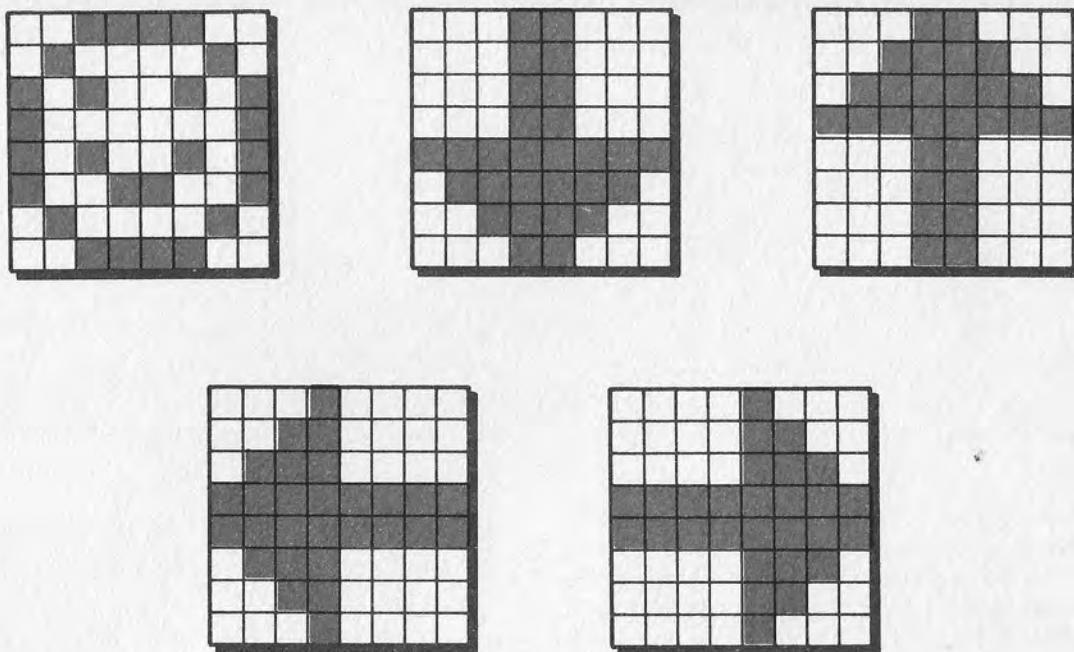


Fig. 3. Otros caracteres que podemos definir.

Una vez que tenemos los nuevos caracteres definidos y almacenados en una cinta de cassette en formato DATA (y a ser posible también en formato interno por si ocurriese algo) nos encontramos con que no sabemos cómo incluirlos en nuestro programa. Para ello sólo tenemos que realizar un MERGE del programa que estamos realizando con el fichero en forma de DATA. Una vez hecho esto veremos que unas líneas DATA se han unido a nuestro programa. Pero ¿qué hacemos con estas líneas? En estas líneas están almacenados los números con la definición de los nuevos caracteres, pero aún no están definidos. Todavía tenemos que introducirlos dentro de la memoria para que puedan ser utilizados. Esto lo podemos conseguir si introducimos estas pocas líneas al principio de programa:

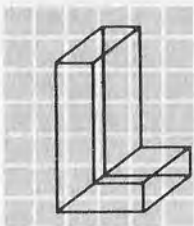
```
100 RESTORE
110 LET DI=BASE (7)
120 LET G=X:REM X=CODIGO ASCII
DEL PRIMER CARACTER DEFINIDO
```

```
130 LET N=Y:REM Y=NUMERO DE CA-
RACTERES DEFINIDOS
140 FOR I=DI+G*8 TO DI+(G+N)*8+7
150 READ A
160 VPOKE I, A
170 NEXT I
```

Con estas pocas líneas introducimos los caracteres definidos en la memoria. Hay que tener en cuenta que para que podamos utilizar estos caracteres tenemos que estar en SCREEN 1. También hay que recordar que si cambiamos de pantalla y luego volvemos a la SCREEN 1, perderemos los caracteres que habíamos definido.

Las letras X e Y que aparecen en el listado no son variables, sino números que nosotros tenemos que introducir. Estas le dicen al programa qué caracteres estamos definiendo. La X le dice al ordenador en qué carácter empezamos la definición y la Y el número de caracteres que hay que leer en las líneas DATA.

■ Programa de control del osciloscopio digital



A circuitería diseñada en el tomo anterior dispone de los componentes necesarios para la captura de las señales analógicas y su almacenamiento en forma digital en una memoria intermedia.

Para seleccionar la velocidad de muestreo, la ganancia del amplificador de entrada y las condiciones de disparo del barrido se incluyeron elementos de control manual, por ser los más asequibles. Además del control de dichas variables, es necesario activar por programa el comienzo de la operación y la transferencia de los datos almacenados a la memoria del ordenador para su tratamiento y presentación.

Para activar las señales de la tarjeta para la toma de muestras es necesario un pequeño programa que ponga las señales de control en los niveles adecuados.

Para la transferencia de datos es necesario un programa que ponga a cero el contador interno de la tarjeta y vaya transfiriendo los datos uno por uno a la memoria del ordenador. En determinadas condiciones de disparo puede resultar conveniente almacenar las muestras de forma continua indefinidamente hasta que se produce la condición de disparo, con objeto de registrar las condiciones que produjeron una determinada forma de la señal.

Para la presentación y tratamiento utilizaremos todas las posibilidades gráficas de nuestra máquina para hacer la información lo más útil posible a nuestro propósito de visualizar los fenómenos observados. Es importante

resaltar que todos tenemos tendencia a interpretar una señal o cualquier fenómeno que se encuentra de forma muestreada de acuerdo con nuestro conocimiento previo. Este condicionamiento sobre la interpretación de las señales muestreadas puede conducir a valoraciones erróneas sobre determinados fenómenos. Es muy importante considerar los resultados del teorema de muestreo en toda su extensión, para comprender que si una señal es de ancho de banda limitado se puede conocer su forma mediante muestras siempre que éstas se tomen con una frecuencia superior a dos veces la frecuencia del armónico más alto existente en la señal. También nos enseña el teorema de muestreo que el error que se observará es la presencia de frecuencias "alias" de la señal real en frecuencias inferiores. Con el ordenador personal podremos mostrar de forma gráfica los resultados de estas afirmaciones.

Hay una situación en la que, por el contrario, se utiliza la creación artificial de alias mediante un muestreo por debajo de las condiciones del teorema, pero "sabiendo lo que se hace". Si la señal que tenemos que observar es repetitiva y de ancho de banda superior a la del osciloscopio de que disponemos, podemos utilizar el siguiente procedimiento: sincronizar con una parte de la señal que se produzca de forma repetitiva y muy estable, y muestrear la señal en puntos de la señal separados de la referencia utilizada para el disparo períodos que se incrementan en cada barrido. El resultado será una señal que tiene la forma de la señal original, pero que se presenta de manera más asequible, por ser de frecuencia inferior. Este procedimiento se ha utilizado durante mucho tiempo y se sigue utilizando para observar señales de muy alta frecuencia. Es el equivalente a observar una rueda en movi-

EL TALLER DEL HARDWARE

miento iluminada con una luz instantánea que se repite a una frecuencia ligeramente inferior a la de rotación de la rueda. Parecerá que la rueda gira a una velocidad igual a la diferencia entre ambas frecuencias. En el caso de que las dos frecuencias coincidan, la rueda parecerá estar quieta. Este fenómeno se denomina estroboscópico y se utiliza con frecuencia para medir la velocidad de rotación de las máquinas, sin contacto directo.

Programa de toma de muestras

El programa presenta un menú de opciones sobre cómo realizar el disparo para comenzar o terminar el barrido, a la frecuencia fijada por la posición del conmutador externo. El ordenador podría determinar el período de muestreo, mediante medidas internas, pero asumiremos que se conoce la posición y, por tanto, la separación entre muestras.

```

10 REM * * * * *
20 REM PROGRAMA DE ACTIVACION DEL OSCILOSCOPIO DIGITAL
50 REM EN EL IBM PC
60 REM * * * * *
61 OPTION BASE 0
62 DEFINT A-Z
65 REM DESCRIPCION DEL PUERTO DE CONTROL:
74 REM * * * * DEFINICION DE ESTADOS DE CONTROL * * * *
75 REPOS0=%H9C: REM NIVELES DE REPOSO AL ARRANCAR
76 ACTIVAR=%H21: REM ACTIVA LA TOMA DE MUESTRAS
77 CONTAR=%H1: REM PULSO DE AVANZAR DIRECCION. INVERTIDO
78 FIN=%B40: REM INDICADOR DE FIN DE MUESTREO
83 INCREMENTAL=%H90: REM LECTURA CON PULSO DE INCREMENTO
84 INCREMENTAH=%H91
86 DIM DATOS(3000)
90 REM * * * * * DEFINICION DE ESCALARES
106 REM * * * * * * * * * * DEFINICION DE PUERTOS
107 PDATOS=%H201: REM DATOS
108 PCONTROL=%H203: REM BYTE DE CONTROL
110 PESTADO=%H203: REM BYTE DE ESTADO
128 KEY OFF
129 CLS
131 PRINT " PROGRAMA DE CONTROL DE OSCILOSCOPIO"
140 PRINT
142 PRINT " VERIFIQUE CONTROLES DE AMPLIFICACION."
143 PRINT " DISPARO Y BASE DE TIEMPOS"
149 PRINT
150 PRINT " SELECCIONE OPCION;"
160 PRINT
170 PRINT " 1.-ACTIVAR MUESTREO"
190 PRINT
200 PRINT " 2.-PRESENTAR RESULTADOS"
210 PRINT
220 PRINT " 3.-TERMINAR"
260 PRINT
270 INPUT "OPCION=":OPCION
280 IF OPCION<1 OR OPCION>3 THEN BEEP: GOTO 129
290 ON OPCION GOSUB 1000,2000,3000
295 IF OPCION=3 THEN END
300 GOTO 129
990 REM * * * * * ACTIVACION DE TOMA DE MUESTRAS
1000 CLS
1010 PRINT "TOMANDO MUESTRAS"
1170 REM ESTABLECE PARAMETROS DE MUESTREO
1200 OUT PCONTROL,ACTIVAR
1210 FOR I=1 TO 10000
1230 A=INP(PESTADO) AND FIN
1240 IF A=1 THEN 1300
1250 NEXT I
1260 PRINT "TERMINO TIEMPO MAXIMO DE TOMA DE MUESTRAS": RETURN
1300 PRINT "FIN DE TOMA DE MUESTRAS"
1310 FOR I=1 TO 2048
1320 OUT CONTROL,INCREMENTAL
1330 OUT CONTROL,INCREMENTAH
1340 OUT CONTROL,INCREMENTAL
1350 DATOS(I)=INP(PDATOS)
1360 NEXT I
1500 RETURN
1990 REM * * * * * MENU DE PRESENTACION * * * *
2000 CLS
2010 REM INCLUIR LOS TIPOS DE PRESENTACION DESEADOS
2100 RETURN
2990 REM * * * * * FUNCION DE PUESTA EN REPOSO * * * *
3000 REM DEJA LA TARJETA EN REPOSO
3010 OUT PCONTROL,REPOSO
3100 RETURN

```

■ Programa de transferencia de datos

Una vez activado el indicador de que el barrido ha terminado, se pueden extraer los datos de la memoria exterior y transferirlos a un vector para su tratamiento y presentación. El programa simplemente consistirá en un bucle de incremento de posición del contador exterior y lectura del valor presente en el puerto de entrada.

La opción de activación de la toma de muestras, cuando termina, realiza automáticamente la transferencia al vector DATOS.

■ Programa de presentación de las señales

La observación de las señales capturadas es el objeto de nuestro experimento. Para facilitar al máximo deberemos disponer de un programa de presentación que utilice óptimamente las posibilidades gráficas de nuestro equipo. El ejemplo que se presenta es una simple muestra de lo que podemos conseguir dedicándole tiempo al problema concreto. Esta es una de las ventajas de utilizar un ordenador: tenemos múltiples soluciones para

cada problema, solamente limitadas por nuestra imaginación.

■ Presentación de las señales tal como se tomaron

El programa presenta las muestras como puntos luminosos o de un color, tomando como eje horizontal el tiempo, que en este caso será el número de la muestra. En el menú de presentación podríamos seleccionar la muestra inicial a presentar y el número de ellas. En el ejemplo la señal se ha almacenado en un fichero y se lee de él cuando se necesita.



Fig. 1. Ejemplo de señal por puntos.

```
5 REM PRESENTACION DE SEÑAL EN FICHERO DE DATOS
10 SCREEN 0,0,0: KEY OFF
30 DIM A(1000)
40 VRES=50: HRES=600: DESPX1=10: DESPY1=30
60 F1$="SEND13"
90 GOSUB 2000: REM LEE LOS DATOS
260 YSCALA=(VRES-1)/ABS(MAXY-MINY)
270 XSCALA=(HRES-1)/N
280 CLS
290 SCREEN 2
295 PRINT " *** REPRESENTACION GRAFICA DE DATOS DIGITALIZADOS ***"
296 PRINT " ESCALA VERTICAL=";YSCALA;" HORIZONTAL=";XSCALA
297 LINE (0,20)-(639,100),1,B
298 FOR I=10 TO 630 STEP 20
299 FOR J=10 TO 80 STEP 10
300 PSET (I,10+J),1
305 NEXT J
310 NEXT I
350 PSET (DESPX1+XSCALA,DESPY1+VRES-1-(A(1)-MINY)*YSCALA)
360 FOR X=2 TO N
384 PSET (DESPX1+X*XSCALA,DESPY1+VRES-1-(A(X)-MINY)*YSCALA)
390 NEXT X
410 LOCATE 16,1:PRINT "VALORES ENTRE ";MINY;" Y ";MAXY;
420 WHILE INKEY$="": WEND
440 SCREEN 0,0,0
500 END
2000 REM LECTURA DE DATOS Y CALCULO MAX Y MIN
2020 OPEN "I",#1,F1$
2030 INPUT #1,N
2040 PRINT "DATOS CONTIENE ";N+1;" VALORES"
2050 INPUT #1,Y
2060 MINY=Y: MAXY=Y
2070 FOR K=1 TO N
2080 INPUT #1,A(K)
2090 IF A(K)>MAXY THEN MAXY=A(K)
2100 IF A(K)<MINY THEN MINY=A(K)
2110 NEXT K
2120 CLOSE
2200 RETURN
```

EL TALLER DEL HARDWARE

El programa realiza el escalado automático, para presentar lo más grande posible a la señal de entrada. Muestra la escala empleada y el desplazamiento, para permitir valorar la señal.

■ Presentación con interpolación

Las señales capturadas con el circuito de muestreo y el conversor analógico/digital contienen la información en los instantes de muestreo de la señal. Con esta información realmente podríamos deducir todo el contenido informativo si se cumplen las condiciones de muestreo. Pero ¿cómo hacer lo más atractiva posible la presentación? En muchos casos es posible mejorar la forma de presentar la señal, en el sentido de hacerla más parecida a la que realmente era, no de desfigurarla para que nos parezca mejor.

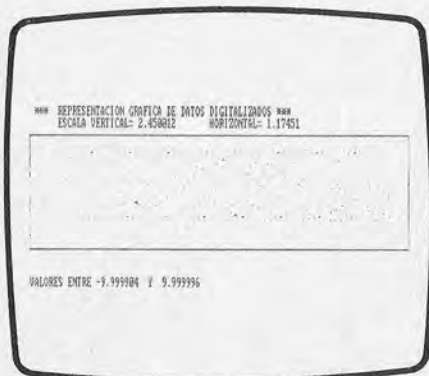


Fig. 2. Señal muestreada con pocas muestras por ciclo.

Supongamos el caso de la figura. La señal era una senoide que se ha muestreado a una frecuencia superior a la mínima requeri-

da y que, por tanto, cumple con las condiciones de muestreo. Sin embargo, la presentación "en crudo" muestra una aparente formación de alias y, lo que es más sorprendente, en un osciloscopio de muestreo es que hay varias señales superpuestas, lo que sabemos que es imposible por el proceso de adquisición de los datos. En un osciloscopio analógico si no está bien sincronizado puede ocurrir que los sucesivos barridos muestren varias partes de la señal que producen las mismas condiciones de disparo.

Si ampliamos la señal, que es la Fig. 1., simplemente mostrando solamente la parte inicial y dejando espacios en blanco, vemos que, efectivamente, se trata de una señal senoidal muestreada a varias veces la componente máxima.

Para evitar esta apariencia podríamos unir los puntos de las muestras con líneas rectas, con lo que, en muchos casos, la señal adopta una apariencia más real. ¿Podríamos hacer más parecida todavía la señal a la original? La respuesta es afirmativa, si consideramos cuál es el interpolador ideal.

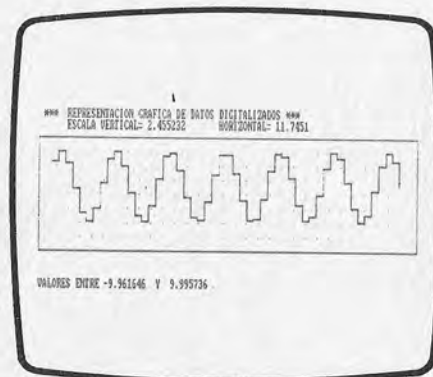


Fig. 3. Interpolador de orden 0.

```

5 REM REPRESENTACION DE SEÑAL POR BARRAS
10 SCREEN 0,0,0: KEY OFF
30 DIM A(1000)
40 VRES=50: HRES=600: DESPX1=10: DESPY1=30
60 F1$="SENOID"
90 GOSUB 2000: REM LEE LOS DATOS
260 YSCALE=(VRES-1)/ABS(MAXY-MINY)
270 XSCALE=(HRES-1)/N
280 CLS
290 SCREEN 2
295 PRINT " *** REPRESENTACION GRAFICA DE DATOS DIGITALIZADOS ***"
296 PRINT "          ESCALA VERTICAL=";YSCALE;"          HORIZONTAL=";XSCALE
297 LINE (0,20)-(639,100),1,B
298 FOR I=10 TO 630 STEP 20
299   FOR J=10 TO 80 STEP 10
300     PSET (I,10+J),1
305   NEXT J
310 NEXT I
350 PSET (DESPX1+XSCALE,DESPY1+VRES-1-(A(1)-MINY)*YSCALE)
360 FOR X=2 TO N

```

```

382 LINE -(DESPX1+X*XSCALA,DESPY1+VRES-1-(A(X-1)-MINY)*YSCALA)
383 LINE -(DESPX1+X*XSCALA,DESPY1+VRES-1-(A(X)-MINY)*YSCALA)
390 NEXT X
410 LOCATE 16,1:PRINT "VALORES ENTRE ";MINY;" Y ";MAXY;
420 WHILE INKEY$="": WEND
440 SCREEN 0,0,0
500 END

2000 REM LECTURA DE DATOS Y CALCULO MAX Y MIN
2020 OPEN "I",#1,F1$
2030 INPUT #1,N
2040 PRINT "DATOS CONTIENE ";N+1;" VALORES"
2050 INPUT #1,Y
2060 MINY=Y: MAXY=Y
2070 FOR K=1 TO N
2080 INPUT #1,A(K)
2090 IF A(K)>MAXY THEN MAXY=A(K)
2100 IF A(K)<MINY THEN MINY=A(K)
2110 NEXT K
2120 CLOSE
2200 RETURN

```

El interpolador es simplemente el instrumento empleado para realizar la operación de asignar valores intermedios de la señal muestreada para su representación. La interpolación puede hacerse "con lápiz y papel", uniendo los valores de las muestras con escalones. En este caso se denominará interpolador de orden cero. Si el número de muestras es pequeño, la señal resultante puede parecer un réplica muy grosera de la original. Se ha sustituido la muestra por una señal rectangular de amplitud igual a la muestra y duración un período.

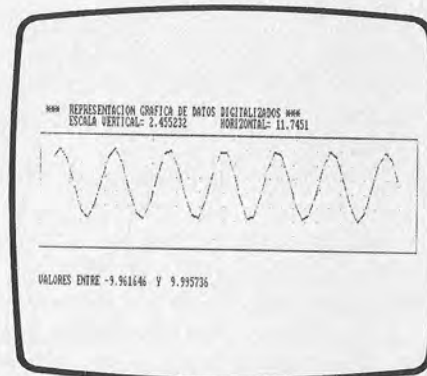


Fig. 4. Interpolador de orden 1.

```

5 REM REPRESENTACION POR LINEAS RECTAS
10 SCREEN 0,0,0: KEY OFF
30 DIM A(1000)
40 VRES=50: HRES=600: DESPX1=10: DESPY1=30
60 F1$="SEND13"
90 GOSUB 2000: REM LEE LOS DATOS
260 YSCALA=(VRES-1)/ABS(MAXY-MINY)
270 XSCALA=(HRES-1)/N
280 CLS
290 SCREEN 2
295 PRINT " *** REPRESENTACION GRAFICA DE DATOS DIGITALIZADOS ***"
296 PRINT "          ESCALA VERTICAL=";YSCALA;"          HORIZONTAL=";XSCALA
297 LINE (0,20)-(639,100),1,B
298 FOR I=10 TO 630 STEP 20
299   FOR J=10 TO 80 STEP 10
300     PSET (I,10+J),1
305   NEXT J
310 NEXT I
350 PSET (DESPX1+XSCALA,DESPY1+VRES-1-(A(1)-MINY)*YSCALA)
360 FOR X=1 TO N
380   LINE -(DESPX1+X*XSCALA,DESPY1+VRES-1-(A(X)-MINY)*YSCALA)
390 NEXT X
410 LOCATE 16,1:PRINT "VALORES ENTRE ";MINY;" Y ";MAXY;
420 WHILE INKEY$="": WEND
440 SCREEN 0,0,0
500 END

2000 REM LECTURA DE DATOS Y CALCULO MAX Y MIN
2020 OPEN "I",#1,F1$
2030 INPUT #1,N
2040 PRINT "DATOS CONTIENE ";N+1;" VALORES"
2050 INPUT #1,Y
2060 MINY=Y: MAXY=Y
2070 FOR K=1 TO N
2080 INPUT #1,A(K)
2090 IF A(K)>MAXY THEN MAXY=A(K)

```

EL TALLER DEL HARDWARE

```

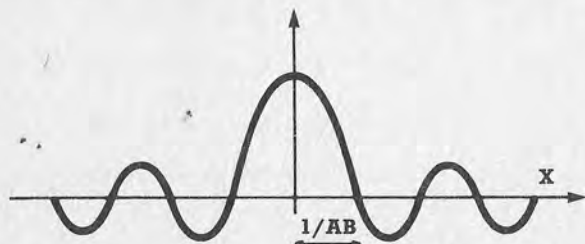
2100 IF A(K) < MINY THEN MINY=A(K)
2110 NEXT K
2120 CLOSE
2200 RETURN

```

Una interpolación más fina consistirá en unir los valores de las muestras con rectas. Requiere el cálculo de la pendiente para su representación. El interpolador se denomina de orden uno, pues el grado del polinomio empleado es uno. El resultado es mejor que el anterior, pero si el número de muestras es pequeño con relación al armónico más alto, resulta bastante deformada la señal. Se ha sustituido la muestra por un triángulo de amplitud igual a la muestra y de base dos periodos.

Siguiendo con polinomios de mayor orden, considerando los puntos adyacentes a los que se unen en cada caso, resultan aproximaciones mejores, aunque requieren mayor número de operaciones.

¿Cuál será el límite teórico para realizar una interpolación perfecta? El resultado es que la forma de onda en los puntos intermedios vendrá definida por la transformada de Fourier correspondiente a la respuesta de un filtro ideal de ancho de banda limitada ante un impulso de la amplitud correspondiente a la muestra. Es decir, si sustituimos cada muestra por la transformada del impulso y sumamos los valores, obtendremos la señal de entrada exactamente. La transformada del impulso es una señal de fórmula: $\text{seno}(x)/x$ y de forma:



Esta señal tiene un máximo en el punto de muestreo y se anula en los demás, por lo que se conservan los valores en cada muestra. Esta curva se atenúa lentamente para valores crecientes de x , por lo que a efectos prácticos se utilizan a veces otras curvas que, dando suficiente aproximación, son más fáciles de calcular y tienen un intervalo de existencia menor. De entre ellas las cúbicas son las más utilizadas, pues presentan una aproximación suave en los extremos de los puntos de muestra.



Fig. 6. Señal interpolada con $\text{seno}(x)/x$.

La señal se ha obtenido con los mismos datos que en los ejemplos anteriores, interpolando con $\text{sen}(x)/x$. Se ha detenido la ejecución del programa a los quince minutos, pues el proceso de interpolación es muy lento. Para una aplicación en condiciones reales convendría utilizar un programa compilado.

Fig. 5. Respuesta a un impulso unidad de un filtro de ancho de banda limitado.

```

5 REM REPRESENTACION CON INTERPOLACION POR SENDIDE
10 SCREEN 0,0,0: KEY OFF
30 DIM A(1000)
40 VRES=50: HRES=600: DESPX1=10: DESPY1=30
60 F1$="SEND13"
70 L=10
80 W=3.141593/L
90 GOSUB 2000: REM LEE LOS DATOS
260 YSCALA=(VRES-1)/ABS(MAXY-MINY)
270 XSCALA=(HRES-1)/N
280 CLS
290 SCREEN 2
295 PRINT " *** REPRESENTACION GRAFICA DE DATOS DIGITALIZADOS ***"
296 PRINT "          ESCALA VERTICAL=";YSCALA;"          HORIZONTAL=";XSCALA
297 LINE (0,20)-(639,100),1,B
298 FOR I=10 TO 630 STEP 20

```

```

299 FOR J=10 TO 80 STEP 10
300 REM PSET (I,10+J),1
305 NEXT J
310 NEXT I
350 PSET (DESPX1+XSCALA,DESPY1+VRES-1-(A(1)-MINY)*YSCALA)
355 K=INT(N/L)
360 FOR X=2 TO N*K
368 G=0
370 FOR J=0 TO N
372 M=W*(X-J*L)
373 IF M=0 THEN G=G+A(J):GOTO 376
374 G=G+A(J)*SIN(M)/M
376 NEXT J
378 PSET (DESPX1+X,DESPY1+VRES-1-(G-MINY)*YSCALA)
386 LOCATE 24,1:PRINT USING "###.###";G:
390 NEXT X
410 LOCATE 16,1:PRINT "VALORES ENTRE ";MINY;" Y ";MAXY;
420 WHILE INKEY$="": WEND
440 SCREEN 0,0,0
500 END

2000 REM LECTURA DE DATOS Y CALCULO MAX Y MIN
2020 OPEN "I",#1,F1$
2030 INPUT #1,N
2040 PRINT "DATOS CONTIENE ";N+1;" VALORES"
2050 INPUT #1,Y
2060 MINY=Y: MAXY=Y
2070 FOR K=1 TO N
2080 INPUT #1,A(K)
2090 IF A(K)>MAXY THEN MAXY=A(K)
2100 IF A(K)<MINY THEN MINY=A(K)
2110 NEXT K
2120 CLOSE
2200 RETURN

```

Otro aspecto interesante a considerar es que después de haber realizado el muestreo, podemos comprimir los datos para su posterior utilización, realizando un "diezmado" que reduzca el número de muestras superfluas. Esta operación puede resultar imprescindible si necesitamos almacenar el resultado de numerosos experimentos, para su posterior tratamiento. Si se cumplen las condiciones de muestreo se podrá reducir significativamente la cantidad de memoria necesaria, hasta una media de 2,5 a 4 muestras por ciclo de la señal de frecuencia más alta.

■ Presentación con comparación

Podemos mostrar varias señales, junto con su comparación, efectuando la resta de las muestras. Es una operación que permitiría calcular la distorsión de un sistema alimentado por un señal senoidal, si a la salida restamos la señal de entrada convenientemente amplificada.

■ Medida del tiempo de subida

En los osciloscopios analógicos los pulsos muy rápidos resultan deformados por las limitaciones de ancho de banda de los amplificadores y la velocidad de escritura de la

pantalla. En un osciloscopio digital, por el contrario, hay una limitación más fuerte: la separación mínima entre muestras. La incertidumbre en la definición de la subida está ligada directamente al mínimo período de muestreo. Esta limitación es también aplicable a cualquier medida de tiempos, pues la incertidumbre aparecerá en los dos puntos donde deseemos marcar el comienzo y final del período.

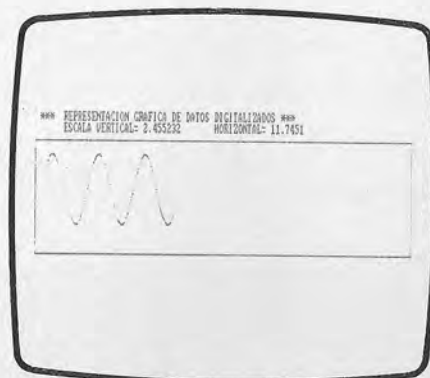
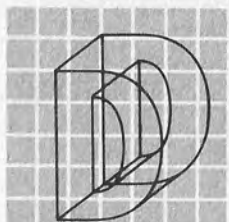


Fig. 7. Incertidumbre del tiempo de subida.

Aquí no resulta aplicable la interpolación senoidal, pues claramente estamos ante una señal que no cumple las condiciones del teorema de muestreo; por tanto, si toda la señal está formada por pulsos, los resultados serán más reales si realizamos interpolación lineal.

NATURALEZA Y TECNOLOGIA

■ Determinación de la concentración de una disolución



ENTRO del capítulo de las disoluciones, uno de los puntos básicos es el del cálculo de las concentraciones en sus diversas formas. Vamos a definir inicialmente cada uno de esos aspectos.

La **Molaridad** es el número de moles de soluto que corresponden a un litro de disolución.

La **Normalidad** es el número de equivalentes-gramo de soluto correspondientes a un litro de disolución.

El **Equivalente-Gramo** de una sustancia es el peso de la misma que reacciona con un gramo de hidrógeno o el cociente entre el mol de la sustancia entre el número de oxidrilos o hidrogeniones que se disocian por molécula de soluto.

Conocida la masa del soluto, su equivalente-gramo y el volumen de la disolución, la normalidad resulta ser:

$$N = \frac{\text{Masa de soluto}}{\text{Volumen de la disolución} \times \text{Equivalente gramo}}$$

La **Molaridad** de una disolución es el número de moles de soluto por litro de disolvente.

La **Fracción Molar** del soluto es la relación entre el número de moles de soluto y el número total de moles de la disolución.

$$FM = \frac{\text{Moles de soluto}}{\text{Moles de soluto} + \text{moles de disolvente}}$$

El número de moles de soluto se calcula a partir de la masa de soluto en la disolución y la masa molecular del soluto

$$NM = \frac{\text{Masa de soluto}}{\text{Masa molecular del soluto}}$$

La masa de disolvente se calcula a partir de su volumen y de su densidad

$$\text{Masa Disolvente} = \text{Volumen del disolvente} \times \text{Densidad del disolvente}$$

El volumen de la disolución es

$$\text{Volumen total} = \frac{\text{Masa de soluto}}{\text{Densidad de soluto}} + \text{Volumen disolvente}$$

```

10 REM *****
20 REM * PROGRAMA : DETERMINACION DE LA *
30 REM * CONCENTRACION DE UNA DISOLUCION *
40 REM * VERSION : 1.0 *
50 REM * PARA IBM-PC Y COMPATIBLES, AMSTRAD *
60 REM * MSX, COMMODORE Y SPECTRUM *
70 REM *****
100 PRINT"DETERMINACION DE LA CONCENTRACION DE UNA DISOLUCION"
110 PRINT"EN SUS DISTINTAS EXPRESIONES: MOLARIDAD, NORMALIDAD,"
120 PRINT"MOLALIDAD Y FRACCION MOLAR"
130 PRINT"INTRODUCCION DE LOS DATOS"
140 PRINT"-----"
150 PRINT"SI DESCONOCE ALGUN DATO PULSE 'N' CUANDO SE LE"
160 PRINT"PREGUNTE EL DATO"
170 PRINT
180 DIM D$(9),D(9)
190 LET I=1
    
```

```

200 INPUT "MASA DE SOLUTO";D$(I)
210 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 200
220 LET D(I)=VAL(D$(I))
230 LET I=I+1
240 INPUT "MASA MOLECULAR DEL SOLUTO";D$(I)
250 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 240
260 LET D(I)=VAL(D$(I))
270 LET I=I+1
280 INPUT "VOLUMEN DE DISOLVENTE";D$(I)
290 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 280
300 LET D(I)=VAL(D$(I))
310 LET I=I+1
320 INPUT "DENSIDAD DEL DISOLVENTE";D$(I)
330 LET D(I)=VAL(D$(I))
340 LET I=I+1
350 INPUT "DENSIDAD DEL SOLUTO";D$(I)
360 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 350
370 LET D(I)=VAL(D$(I))
380 LET I=I+1
390 INPUT "MASA MOLECULAR DEL DISOLVENTE";D$(I)
400 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 390
410 LET D(I)=VAL(D$(I))
420 LET I=I+1
430 INPUT "DENSIDAD DE LA DISOLUCION";D$(I)
440 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 430
450 LET D(I)=VAL(D$(I))
460 LET I=I+1
470 INPUT "VOLUMEN DE LA DISOLUCION";D$(I)
480 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 470
490 LET D(I)=VAL(D$(I))
500 LET I=I+1
510 INPUT "VOLUMEN DE SOLUTO";D$(I)
520 IF VAL(D$(I))=0 AND D$(I)<>"N" THEN GOTO 510
530 LET D(I)=VAL(D$(I))
540 LET NS=D(1)/D(2)
550 LET MD=D(3)/D(4)
560 LET VS=D(1)/D(5)
570 LET WS=D(3)+D(9)
580 PRINT NS;"MOLES DE SOLUTO"
590 PRINT MD;"MASA DE DISOLVENTE"
600 PRINT VS;"VOLUMEN DE SOLUTO"
610 PRINT WS;"VOLUMEN TOTAL DE LA DISOLUCION"
620 IF D$(8)="N" THEN GOTO 650
630 LET MR=1000*NS/D(8)
640 GOTO 660
650 LET MR=1000*NS/WS
660 INPUT "¿CUANTO VALE LA MOLARIDAD DE LA DISOLUCION?";VM
670 IF VM<>MR THEN GOTO 690
680 PRINT "RESPUESTA CORRECTA":GOTO 700
690 PRINT "RESPUESTA INCORRECTA. LA RESPUESTA CORRECTA ES";MR;"M"
700 INPUT "INTRODUZCA EL NUMERO DE OXIDRILOS O HIDROGENIONES QUE SE PRODUCE POR
    MOLECULA";HH
710 LET NR=MR*HH
720 INPUT "¿CUANTO VALE LA NORMALIDAD DE LA DISOLUCION?";VN
730 IF VN<>NR THEN GOTO 750
740 PRINT "RESPUESTA CORRECTA":GOTO 760
750 PRINT "RESPUESTA INCORRECTA. LA RESPUESTA CORRECTA ES";NR;"N"
760 LET ML=NS*1000/(D(3)*D(4))
770 INPUT "¿CUANTO VALE LA MOLALIDAD DE LA DISOLUCION?";VL
780 IF ML<>VL THEN GOTO 800
790 PRINT "RESPUESTA CORRECTA":GOTO 810
800 PRINT "RESPUESTA INCORRECTA. LA RESPUESTA CORRECTA ES";ML;"M1"
810 LET FM=NS/(NS+(MD/D(7)))
820 INPUT "¿CUANTO VALE LA FRACCION MOLAR DE LA DISOLUCION?";FV
830 IF FM<>FV THEN GOTO 850
840 PRINT "RESPUESTA CORRECTA":GOTO 860
850 PRINT "RESPUESTA INCORRECTA. LA RESPUESTA CORRECTA ES";FM;"%"
860 END

```

PROBLEMAS MATEMATICOS

■ Circunferencia definida por tres puntos

A partir de las coordenadas cartesianas de tres puntos, P_1 , P_2 y P_3 , se puede obtener la

circunferencia que pase por esos tres puntos. El programa da como solución las coordenadas del centro de la circunferencia y su radio.

Siendo (x_1, y_1) , (x_2, y_2) , (x_3, y_3) las coordenadas específicas de los puntos P_1 , P_2 , P_3 , la

pendiente de la recta que une P_1 con P_2 resulta de

$$\frac{Y_2 - Y_1}{X_2 - X_1}$$

la pendiente de todas las perpendiculares a esa recta es

$$-\frac{X_2 - X_1}{Y_2 - Y_1}$$

la ecuación de la mediatriz del segmento P_1, P_2 es

$$y = \frac{Y_1 + Y_2}{2} - \frac{X_2 - X_1}{Y_2 - Y_1} \left(x - \frac{X_1 + X_2}{2} \right)$$

la ecuación de la mediatriz del segmento P_1, P_3 es

$$y = \frac{Y_1 + Y_3}{2} - \frac{X_3 - X_1}{Y_3 - Y_1} \left(x - \frac{X_1 + X_3}{2} \right)$$

Estas dos ecuaciones se pueden poner en la forma general de una ecuación de una recta de la siguiente forma

$$y = K_2x + H_2$$

$$y = K_3x + H_3$$

donde

$$K_2 = -\frac{X_2 - X_1}{Y_2 - Y_1} \quad y \quad K_3 = -\frac{X_3 - X_1}{Y_3 - Y_1}$$

$$H_2 = \frac{Y_1 + Y_2}{2} + \frac{X_2^2 - X_1^2}{2(Y_2 - Y_1)}$$

$$H_3 = \frac{Y_1 + Y_3}{2} + \frac{X_3^2 - X_1^2}{2(Y_3 - Y_1)}$$

Resolviendo el sistema de ecuaciones se obtienen las coordenadas del centro de la circunferencia

$$X = \frac{H_3 - H_2}{K_2 - K_3} \quad Y = \frac{K_3 H_2 - K_2 H_3}{K_3 - K_2}$$

A partir de las coordenadas del centro se obtiene el radio de la circunferencia

$$R = \sqrt{(X_1 - X)^2 + (Y_1 - Y)^2}$$

```

10 REM *****
20 REM * PROGRAMA : CIRCUNFERENCIA DEFINIDA *
30 REM * POR TRES PUNTOS *
40 REM * VERSION : 1.0 *
50 REM * PARA IBM-PC Y COMPATIBLES, AMSTRAD, *
60 REM * MSX, COMMODORE Y SPECTRUM *
70 REM *****
90 DIM X(3),Y(3)
100 PRINT"CIRCUNFERENCIA DEFINIDA POR TRES PUNTOS"
110 PRINT"INTRODUZCA LA COORDENADAS DE LOS TRES PUNTOS"
120 FOR I=1 TO 3
130 PRINT"COORDENADA X";I;"=";
140 INPUT X(I)
145 PRINT"COORDENADA Y";I;"=";
147 INPUT Y(I)
150 NEXT I
160 LET K2=-(X(2)-X(1))/(Y(2)-Y(1))
170 LET K3=-(X(3)-X(1))/(Y(3)-Y(1))
180 LET D=K3-K2
190 IF D=0 THEN PRINT"LOS PUNTOS ESTAN ALINEADOS. NO HAY SOLUCION":END
200 LET H2=.5*(Y(1)+Y(2))+((X(2)^2)-(X(1)^2))/(Y(2)-Y(1))
210 LET H3=.5*(Y(1)+Y(3))+((X(3)^2)-(X(1)^2))/(Y(3)-Y(1))
220 LET X=(H2-H3)/D
230 LET Y=(K3*H2-K2*H3)/D
240 LET R=SQR(((X(1)-X)^2)+((Y(1)-Y)^2))
250 PRINT"LA SOLUCION ES"
260 PRINT"COORDENADAS DEL CENTRO : X=";X;" Y=";Y
270 PRINT"RADIO : R=";R

```

```

500 REM *****
510 REM * MODIFICACIONES PARA PINTAR EL CIRCULO *
520 REM * PARA SPECTRUM *
530 REM *****
540 IF X-R<0 OR X+R>255 OR Y-R<0 OR Y+R>175 THEN GOTO 560
550 CIRCLE X,Y,R
560 PRINT"EL DIBUJO SE SALE DE LOS LIMITE DE LA PANTALLA"

```

```

500 REM *****
510 REM * MODIFICACIONES PARA PINTAR EL CIRCULO *
520 REM * PARA IBM-PC Y COMPATIBLES Y MSX *
530 REM *****
540 IF X-R<0 OR X+R>640 OR Y-R<0 OR Y+R>480 THEN GOTO 570
550 SCREEN 2
560 CIRCLE (X,Y),R
570 PRINT"EL DIBUJO SE SALE DE LOS LIMITES DE LA PANTALLA"

```

SOCIEDAD

■ La generación del 27

Las características más importantes de este grupo de poetas son las siguientes:

Convivencia en la Residencia de Estudiantes de Madrid y en el Centro de Estudios Históricos.

Celebración en 1927 del centenario de Góngora en el Ateneo de Sevilla.

Participación en la *Revista de Occidente*, en la *Gaceta Literaria*, en el *Litoral*, de Málaga...

Tendencia al equilibrio entre lo intelectual y lo sentimental, entre una concepción cuasi mística de la poesía y una lucidez rigurosa en la elaboración del poema, entre la pureza estética y la autenticidad humana, entre lo minoritario y "la inmensa compañía" y entre lo universal y lo español.

Asimilación de lo anterior, respeto a la tradición y renovación de la lírica española.

El programa cuyo listado se ofrece a continuación preguntará a qué autor de la generación pertenece la obra que aparezca en pantalla.

```

10 REM *****
20 REM * PROGRAMA : GENERACION DEL 27 *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * AMSTRAD Y COMMODORE *
60 REM *****
100 DIM A$(10),P$(51),P(51)
110 FOR I=1 TO 10
120 READ A$(I)
130 NEXT I
140 FOR I=1 TO 51
150 READ P$(I)
160 READ P(I)
170 NEXT I
180 LET R=0
190 FOR I=1 TO 10
200 LET N=INT(RND(1)*51)+1
210 PRINT"¿A QUE AUTOR PERTENECE LA OBRA ";P$(N);"?
220 INPUT A$
230 IF A$=A$(P(N)) THEN PRINT"CORRECTO":LET R=R+1:GOTO 260
240 PRINT"RESPUESTA INCORRECTA"
250 PRINT"LA RESPUESTA CORRECTA ES";A$(P(N))
260 NEXT I
270 PRINT"HA TENIDO";R;"ACIERTOS"
280 END
1000 DATA PEDRO SALINAS,JORGE GUILLEN,GERARDO DIEGO
1010 DATA DAMASO ALONSO,VICENTE ALEIXANDRE
1020 DATA FEDERICO GARCIA LORCA,RAFAEL ALBERTI
1030 DATA LUIS CERNUDA,EMILIO PRADOS,MANUEL ALTOLAGUIRRE
1040 DATA PRESAGIOS,1,SEGURO AZAR,1,FABULA Y SIGNO,1
1050 DATA LA VOZ A TI DEBIDA,1,RAZON DE AMOR,1,CONFIANZA,1
1060 DATA CANTICO,2,CLAMOR,2,HOMENAJE,2,IMAGEN,3
1070 DATA MANUAL DE ESPUMAS,3,POEMAS ADREDE,3
1080 DATA VERSOS HUMANOS,3,BORIA,3,ALONDRA DE VERDAD,3
1090 DATA EL VIENTO Y EL VERSO,4,HIJOS DE LA IRA,4
1100 DATA HOMBRE Y DIOS,4,ESPADAS COMO LABIOS,5
1110 DATA MUNDO A SOLAS,5,LA DESTRUCCION O EL AMOR,5
1120 DATA SOMBRA DEL PARAISO,5,HISTORIA DEL CORAZON,5
1130 DATA POEMAS DE LA CONSUMACION,5,LIBRO DE POEMAS,6
1140 DATA CANCIONES,6,POEMA DEL CANTE JONDO,6
1150 DATA ROMANCERO GITANO,6,POETA EN NUEVA YORK,6
1160 DATA LLANTO POR IGNACIO SANCHEZ MEJIAS,6
1170 DATA MARINERO EN TIERRA,7,LA AMANTE,7,EL ALBA DE ALHELI,7

```

```

1180 DATA CAL Y CANTO,7,SOBRE LOS ANGELES,7,EL ANGEL DE LA IRA,7
1190 DATA TORO EN EL MAR,7,POEMAS DEL DESTIERRO,7
1200 DATA LA REALIDAD Y EL DESEO,8,PERFIL DEL AIRE,8
1210 DATA UN RIO UN AMOR,8,LOS PLACERES PROHIBIDOS,8
1220 DATA INVOCACIONES,8,CON LAS HORAS CONTADAS,8
1230 DATA TIEMPO A CUERPO PERSEGUIDO,9,LLANTO EN LA SANGRE,9
1240 DATA JARDIN CERRADO,9,LAS ISLAS INVITADAS,10,EJEMPLO,10
1250 DATA SOLEDADES JUNTAS,10,POEMAS DE AMERICA,10

```

Modificaciones para otros equipos

SPECTRUM

LINEA 100 DIM A\$(10,30),P\$(51,30),P(51)

PARA LOS MAS JOVENES

■ Animales invertebrados

Como complemento al programa sobre animales vertebrados, en esta ocasión hemos querido dar un repaso a tus conocimientos sobre la fauna invertebrada. Antes de nada vamos a repasar ciertos conceptos.

Los **Artrópodos** engloban a los **Insectos**, **Arácnidos**, **Miriápodos** y **Crustáceos**.

Los **Insectos** tienen el cuerpo cubierto de quitina, están formados de anillos, tienen dos antenas, llevan seis patas y alas (cuatro o dos, a veces ninguna), respiran por tráqueas, nacen de huevos y experimentan metamorfosis.

Los **Arácnidos** tienen su cuerpo dividido en cefalotórax y abdomen, llevan quelíceros y pedipalpos, tienen ocho patas, respiran por tráqueas, no tienen antenas y nacen de huevos.

Los **Miriápodos** están cubiertos de quitina, tienen el cuerpo formado por anillos, con uno o dos pares de patas en cada uno, tienen respiración traqueal y experimentan mudas y no metamorfosis.

Los **Crustáceos** tienen caparazón o costura dura que los protege, experimentan mudas para crecer, llevan dos pares de antenas y de ordinario diez patas, son acuáticos, respiran por branquias y nacen de huevos.

Los **Moluscos** tienen el cuerpo blando sin formar anillos, pueden tener concha con una o dos valvas, se mueven por un pie musculoso y nacen de huevos. Atendiendo a su respiración pueden dividirse en respiración traqueal o respiración pulmonar.

Los **Equinodermos** tienen el cuerpo cubierto de placas calizas con púas, tienen pies ambulacrales y pueden dividirse por su forma en estrellados y esféricos.

Los **Gusanos** son animales delgados y alargados, tienen el cuerpo blando y con frecuencia formado por anillos, carecen de apéndices y de patas articuladas, pero llevan cerdas para caminar, algunos producen enfermedades, nacen de huevos y se dividen en **Anélidos**, **Plantelmintos** y **Nematelmintos**.

Los **Pólipos** tienen el cuerpo en forma de saco, brazos alrededor de la boca, poseen células urticantes y suelen ser marinos. Las **Eponjas** tienen el cuerpo perforado con agujeros que les atraviesan el cuerpo, suelen ser marinas y con forma de saco.

Los **Protozoos** son animales unicelulares, viven en el agua o en las plantas verdes, se enquistan por falta de humedad, unos son inofensivos y otros producen enfermedades y los más conocidos son **Infusorios** y **Flagelados**.

El programa te preguntará a qué tipo y a qué clase pertenecen diez animales elegidos al azar.

```

10 REM *****
20 REM * PROGRAMA:ANIMALES INVERTEBRADOS *
30 REM * VERSION : 1.0 *
40 REM * PARA IBM-PC Y COMPATIBLES, MSX, *
50 REM * COMMODORE Y AMSTRAD *
60 REM *****
100 PRINT"CON ESTE PROGRAMA PODRAS VERIFICAR"
110 PRINT"TUS CONOCIMIENTOS DE LOS ANIMALES "
120 PRINT"INVERTEBRADOS. PARA ELLO, EL ORDENADOR"
130 PRINT"TE PREGUNTARA QUE TIPO DE ANIMAL ES"

```

```

140 PRINT"EL QUE APARECE EN PANTALLA Y A QUE "
150 PRINT"CLASE PERTENECE. ASI, POR EJEMPLO,"
160 PRINT"SI APARECE EN PANTALLA 'MOSCA' A LA"
170 PRINT"PREGUNTA DE TIPO TENDRAS QUE RESPONDER"
180 PRINT"ARTROPODO Y A LA PREGUNTA DE CLASE "
190 PRINT"RESPONDERAS INSECTOS."
200 PRINT
210 DIM T$(6),C$(15),A$(36),A(36,2)
300 FOR I=1 TO 6
310 READ T$(I)
320 NEXT I
330 FOR I=1 TO 15
340 READ C$(I)
350 NEXT I
360 FOR I=1 TO 36
370 READ A$(I)
380 READ A(I,1)
390 READ A(I,2)
400 NEXT I
480 LET R=0
490 FOR I=1 TO 10
500 N=INT(RND(1)*36)+1
510 PRINT"%DE QUE TIPO ES EL SIGUIENTE ANIMAL? ";A$(N)
520 INPUT R$
530 IF R$=T$(A(N,1)) THEN PRINT"BIEN":R=R+1:GOTO 550
540 PRINT"RESPUESTA INCORRECTA"
550 PRINT"%A QUE CLASE DE ";T$(A(N,1));" PERTENECE?"
560 INPUT R$
570 IF R$=C$(A(N,2)) THEN PRINT"BIEN":R=R+1:GOTO 590
580 PRINT"RESPUESTA INCORRECTA"
590 PRINT"LA RESPUESTA COMPLETA ES ";A$(N);" ";T$(A(N,1));" ";C$(A(N,2))
600 NEXT I
610 PRINT"HAS TENIDO ";R/2;" ACIERTOS"
900 DATA ARTROPODOS,MOLUSCOS,EQUINODERMOS,GUSANOS
910 DATA POLIPOS Y ESPONJAS,PROTOZOOS
920 DATA INSECTOS,ARACNIDOS,MIRIAPODOS,CRUSTACEOS
930 DATA PULMONARES,BRANQUIALES,ESTRELLADOS,ESFERICOS
940 DATA ANELIDOS,PLATELMINTOS,NEMATELMINTOS
950 DATA POLIPOS,ESPONJAS,INFUSORIOS,FLAGELADOS
1000 DATA SALTAMONTES,1,1,MARIPOSA,1,1,ABEJA,1,1,AVISPA,1,1
1010 DATA MOSCA,1,1,MOSQUITO,1,1,HORMIGA,1,1
1020 DATA ARANA,1,2,ESCORPION,1,2,GARRAPATA,1,2
1030 DATA ESCOLOPENDRA,1,3,CIEMPIES,1,3
1040 DATA CANGREJO,1,4,LANGOSTA,1,4,LANGOSTINO,1,4
1050 DATA GAMBA,1,4,CIGALA,1,4,CARACOL,2,5,BABOSA,2,5
1060 DATA MEJILLON,2,6,ALMEJA,2,6,OSTRA,2,6,PULPO,2,6
1070 DATA ESTRELLA,3,7,OFIURA,3,7,ERIZO,3,8,HOLOTURIA,3,8
1080 DATA LOMBRIZ,4,9,SANGUIJUELA,4,9,TENIA,4,10,TRIQUINA,4,11
1090 DATA HIDRA,5,12,ESPONJA,5,13,PARAMECIO,6,14,VORTICELA,6,14
1100 DATA TRYPANOSOMA,6,15
    
```

Modificaciones para otros equipos

SPECTRUM

```

210 DIM T$(6,15); DIM C$(15,15); DIM
A$(36,15); DIM A(36,2)
    
```

PEQUEÑA HISTORIA DE LA INFORMATICA

■ Los calculadores diseñados en los laboratorios Bell



ARA muchas personas (los humanos nunca nos ponemos de acuerdo sobre estas cosas), el primer calculador binario que se creó fue obra de otro matemático, George Stibitz, que trabajaba en los laboratorios de la gran compañía Bell. En su trabajo diario, Stibitz utilizaba muchos tipos de relés y la simplicidad de comportamientos típica de este elemento (todo o nada) le sugirió que sería muy útil en la construcción de calculadoras, ya que evitaba situaciones ambiguas. Como las máquinas ya no eran simplemente mecánicas, la velocidad tampoco sufriría grandes recortes... Stibitz, por su cualidad de matemático, había estudiado las teorías de otro matemático irlandés desconocido en aquel momento, George Boole (1815-1854). Boole expresaba con una enorme simplicidad los problemas lógicos mediante un álgebra que él mismo había pensado. Aunque Boole pasó por la vida como un ciudadano más, sin mayor trascendencia, la informática y otras ciencias relacionadas con las Matemáticas lo han hecho vigente, devolviéndole la importancia que se merece.

En la lógica booleana se consideran dos dígitos, uno y cero, que representan conjuntos con elementos, o vacíos. Además de los dígitos, Boole considera operaciones, como OR-O o AND-Y, que corresponden, respectivamente, a la adición y multiplicación booleanas.

Este sistema era idóneo para las calculadoras, ya que los unos y ceros podían representarse mediante contactos, abiertos o cerra-

dos, sin ningún tipo de situaciones intermedias.

A pesar del interés que despertó la lógica binaria, sobre todo aplicada al diseño de máquinas de calcular, seguían existiendo, y se siguieron fabricando máquinas que utilizaban sistemas decimales, que, por otro lado, tenían algunas ventajas, aunque el tiempo ha demostrado que eran menores que los inconvenientes.

Cuando Stibitz se decidió a crear su máquina, en el verano de 1937, la mayoría de las máquinas existentes utilizaban sistema decimal. Stibitz era un amante de la experimentación y, ocupando su tiempo libre en aquello que más le apasionaba, había ido recogiendo elementos que posiblemente le sirvieran en sus «trabajos», algunos relés sacados de entre los desechados, piezas metálicas de aquí y allá y un par de bombillas. Estas bombillas servían de indicadores. Si la suma suponía un 1, se encendían, permaneciendo apagadas si lo obtenido era un cero.

A partir de ese importante momento, la familia de máquinas que disponen de unidades aritmético-lógicas que utilizan el sistema binario y las memorias con dos estados estables ha ido creciendo hasta las cotas que hoy en día conocemos. (Cualquiera ha podido constatar que existen relojes-calculadora por menos de 700 pesetas.) Imagine el lector con estas premisas el número de calculadoras que han podido fabricarse desde entonces.

Stibitz no dispuso de elementos caros para fabricar su primer prototipo, pero eso no quiere decir, ni mucho menos, que lo hubiera dado por las correspondientes 700 pesetas. Y así fabricó máquinas que multiplicaban y dividían. Su apreciación del coste de un calculador completo era de unos 50.000 dólares. Naturalmente, en esas fechas esa cantidad era

PEQUEÑA HISTORIA DE LA INFORMATICA

astronómica y, por tanto, el invento de Stibitz podía muy bien no valer para nada. Pero Stibitz decidió no perder su inspiración y convenció a sus superiores en el laboratorio para que le financiaran otra máquina, más compleja, que realizara automáticamente cálculos complicados. En el diseño y fabricación fue ayudado por otro ingeniero del laboratorio, Sam Williams. A principios de 1939, el grupo comenzó a trabajar en el Complex Calculator, que más tarde fue rebautizado con el nombre de Bell Labs Relay Computer Model I, o Bell Telephone Labs Computer Model I (BTL Model I). La máquina llevaba más de cuatrocientos relés, de los normales de teléfonos, y tenía una pequeña memoria. Los datos se introducían a través de un teletipo. La máquina era bastante completa, aunque tenía limitaciones. Lo fundamental es que, aunque trabajaba con números complejos, sólo realizaba las cuatro operaciones básicas. Sin embargo, su mérito consistió en ser el primer calculador binario realmente utilizado. (Aunque sobre estos temas, de quién es el primero, siempre se encuentran versiones controvertidas.)

Como acabamos de decir, al poco tiempo de ser creado el Model I, el grupo de diseñadores no estaba satisfecho de su labor, y reemprendió la tarea de crear otra máquina que superara las deficiencias de la primera. La máquina se llamó naturalmente Bell Labs Model 2 (BTL Model 2) al menos calculaba polinomios. También algo más tarde este nuevo modelo cambió de nombre, y pasó a ser conocido como Relay Interpolator (Interpolador de Relés). El cálculo de polinomios se realizaba ejecutando secuencias de instrucciones que podían introducirse mediante teletipos, o grabadas en tiras de papel. La máquina era magnífica, y podía realizar multitud de cálculos, aunque no llegaba a ser un verdadero calculador universal.

Los restantes modelos, el modelo 3 y el 4, fueron mejoras del modelo 2, pero tampoco resolvieron totalmente sus problemas. El primer calculador universal también perteneció a la serie anterior, pero había conseguido superar finalmente sus limitaciones, constituyendo una máquina de uso general. Como el Harvard Mark II (primer «ordenador»), era enorme. Para efectuar cálculos (con siete cifras decimales) tardaba un tercio de segundo en las sumas, un segundo para las multiplicaciones y algo más de dos segundos para las divisiones. Para ello utilizaba unos 10.000 relés, pesaba casi una tonelada y ocupaba una planta de

¿Sabía usted que...

La NSA (National Security Agency) es un organismo dependiente del Ministerio norteamericano de Defensa, «especializado en actividades altamente técnicas», relacionadas con la vigilancia de los Estados Unidos. Fue fundada en 1952, aunque apenas si se notó su presencia, a nivel del ciudadano de la calle. Sin embargo, posee un centro informático de los más avanzados y potentes del mundo. Entre sus ordenadores dispone de varios IBM 3033, Cray, etc., son capaces de realizar de 150 a 200 millones de cálculos por segundo. Las bases de datos a que pueden acceder son impensables, las memorias transfieren 320 millones de palabras por segundo (correspondientes a unos dos mil quinientos libros de unas trescientas páginas). Las impresoras IBM, muy especializadas, imprimen 22.000 líneas por minuto. Pero el centro de cálculo no es un organismo estático. No basta con lo obtenido, hay que mejorar las características de los ordenadores, sobre todo su velocidad y potencia. Se han comenzado a realizar trabajos de investigación con rayos láser, y los resultados parecen prometedores.

más de 100 metros cuadrados. Sin embargo, salió a la luz dos años después del Mark II (Harvard IBM), máquina enorme y completísima, de la que hemos hablado someramente en otras ocasiones.

El laboratorio de la Bell siguió sus actividades, después de los logros conseguidos, hasta llegar al Model 5, en 1946. El ambiente era triunfal, y siguieron fabricando máquinas, de las que se sentían enormemente satisfechos, aunque con el paso del tiempo no ha quedado demostrada su enorme eficacia. Las máquinas eran utilizadas con fines científicos, entre otros, en el propio laboratorio.

Sin embargo, el «invento» más relevante del equipo que trabajaba en el laboratorio ocurrió cuando pensaron en unir dos teletipos a la máquina, para que ésta fuera realizando cálculos dando servicio a ambos teletipos. Más tarde, comprobaron que la máquina podía aceptar un tercer teletipo. Esta pequeña hazaña constituyó el primer «prototipo» de sistema de comunicaciones. El sistema adoptado es lo que hoy llamaríamos FIFO (First in, first out—primero en entrar, primero en salir—); es decir, nada más pedir servicio el primer teletipo accede al uso de la máquina, al finalizar los cálculos, se concede el acceso al teletipo que más ha esperado, y así sucesivamente.

En sus primeros años de trabajo en el laboratorio de la Bell, Stibitz inventó el sistema de coma flotante para los números, avance muy importante para poder procesar números que de otro modo resultarían muy grandes. El sistema consistía simplemente en dividir dichos números por potencias de diez, almacenando la operación realizada. A continuación, se realizaban los cálculos con el número simplificado, para terminar deshaciendo la simplificación, llevando a cabo la operación contraria a la que se había almacenado. La técnica fue muy utilizada en los laboratorios de la Bell, para todas las máquinas que empleaban, aunque, no sabemos por qué, tuvo muy poca repercusión en otras máquinas, y no se instauró hasta varios años después, en otros calculadores universales. Hoy en día, la práctica totalidad de las calculadoras funcionan con coma flotante.

En 1940 se produjo un acontecimiento brillante y esperanzador. La American Mathematical Society estaba reunida en Dartmouth College, New Hampshire. Como demostración de las posibilidades combinadas de calculador universal/red de teletipos, Stibitz conectó un teletipo a uno de los calculadores universales de la compañía, en Nueva York. De este modo, introduciendo los datos desde un lugar remoto cualquiera, se podían obtener los resultados de los cálculos efectuados por la máquina en tiempos realmente pequeños, es decir, minutos. Con este sistema, los participantes podían enviar cálculos a la máquina y re-

¿Sabía usted que...

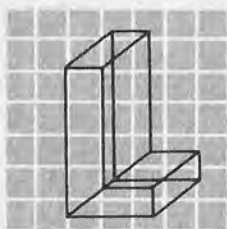
La IBM es un cliente importante de la propia IBM.

Efectivamente, la compañía invierte grandes cantidades de dinero en proceso de datos. Además, teniendo en cuenta que el precio de sus productos es un 40 % más barato, la cifra total es escalofriante. El número de personas que trabajan en proceso de datos es también enorme (representa aproximadamente el 5 % del personal total de la compañía). En resumen, la propia IBM es el segundo cliente en importancia de la propia compañía, siendo aventajado únicamente por el Gobierno de los Estados Unidos de América.

cibir las respuestas, e incluso recibir datos y problemas desde otros puntos (otras ciudades, e incluso desde otros países). En aquellas fechas, el acontecimiento causó gran conmoción, aunque, como hemos visto en otros temas, los avances se sucedían a una velocidad vertiginosa para la época.

Como puede ver el lector, aunque los algoritmos para la suma, multiplicación y división datan de tiempos prehistóricos, apenas si se avanzó en mejorar estos cálculos tan sencillos hasta el siglo XVII. Y a partir de entonces, lentamente al principio, y a velocidad fulgurante en nuestro siglo, se ha avanzado en la resolución de algoritmos complicadísimos.

■ Ordenadores especializados en el manejo de bases de datos



AS previsiones para el futuro indican que las bases de datos son y serán cada vez más un elemento indispensable en la informatización de la sociedad. Parece evidente que es un método muy eficaz de aprovechar

los conocimientos de especialistas de primera fila en cada una de las materias de una forma sencilla y sobre todo rápida. La difusión de las bases de datos existentes, así como de los procedimientos para acceder a ellas, es fundamental, sobre todo en los casos de bases de datos extranjeras.

La posibilidad de disponer de esta información crea flujos «de saber» a nivel internacional, lo que redundará, desde luego, en un avance de las técnicas muy superior al producido hasta ahora.

Pero estas bases de datos deben poder ser manejadas por sistemas eficaces. Lo mejor es disponer de ordenadores de bases de datos (DB Computers) diseñados especialmente para hacerlo. Actualmente, existen ordenadores que cumplen estas condiciones, ya comercializados, y también prototipos experimentales, que emplean técnicas más avanzadas. El software es también un elemento muy importante, que va depurándose con el tiempo.

Hasta el momento, se puede considerar que ha habido un cierto descuido o falta de interés en el diseño de herramientas específicas y la elección de aquellas metodologías más apropiadas para el análisis de la arquitectura de los nuevos sistemas de bases de datos. Es necesario, pues, un análisis cuidadoso para un diseño eficaz. Sin él, será imposible

evaluar ni la organización ni otras características fundamentales del hardware de las máquinas específicas de manejo de bases de datos. En unos tiempos tan marcados por la eficacia no podemos basarnos exclusivamente en «experiencias» o «intuiciones» de diseñadores. Además, en la práctica, entre gran número de diseños, se han producido muchos malos (y que por ello no han podido ser implementados, dificultando y, en suma, haciendo inoperante el equipo), y algunos buenos, aunque problemas de tipo económico han evitado la difusión de tales sistemas.

En fechas actuales, existen ordenadores de bases de datos (DB computers) magníficamente diseñados, que cumplen sus expectativas perfectamente. Sin embargo (y aunque no parezca muy creíble), sus características siguen conservándose ocultas, como los artesanos guardaban celosamente sus pequeños trucos, o los operarios egipcios de la Casa de los Muertos, sus fórmulas de conservación de los cadáveres embalsamados.

Pero olvidemos a los muertos, y pasemos a algo muy vivo. A pesar del oscurantismo existente, hay técnicas de evaluación de las características de los ordenadores bastante fiables. Una de las metodologías existentes es el Benchmarking, que evalúa tanto las máquinas como las propias bases de datos. Pueden utilizarse también otras metodologías (monitores de hardware o de software, baterías de tests, etc.), pero el objetivo común es el análisis del producto (o prototipo), llegando incluso hasta a evaluar su diseño.

Como estamos viendo, los DB computers no son simplemente grandes ordenadores (main frames). La diferencia fundamental estriba en que los grandes ordenadores son de propósito general, mientras que los DB son de uso específico. Por ejemplo, los grandes ordenadores han sido diseñados poniendo un én-

fasis especial en las velocidades de proceso de los programas almacenados en su memoria. Sin embargo, los DB computers no tienen por qué tener programas almacenados en su memoria. Por el contrario, son máquinas diseñadas especialmente para realizar transacciones de entrada/salida de datos de forma intensiva, a diferencia de los grandes ordenadores que computan programas también de forma intensiva.

Otra diferencia fundamental entre ambos tipos de ordenador radica en que los recursos físicos del sistema son controlados por el sistema operativo del ordenador, mientras que en los sistemas de bases de datos son los que controlan la propia base de datos, es decir, recursos físicos y lógicos, ya que la base de datos está formada por unos elementos «raíz» y unas relaciones semánticas (o lógicas) entre dichos elementos.

El esquema fundamental de un DB computer está indicado en la figura. El ordenador DB actúa como máquina «dedicada», es decir, máquina de diseño específico y utilizada con una finalidad concreta, frente a uno o varios ordenadores de uso general (frontend). Al conjunto de todos los ordenadores (frontend) que acceden al DB computer lo hemos designado como «host». Los programas del usuario residen en estos ordenadores «host», que son los que procesan, pudiendo utilizar el ordenador DB como otro más de los recursos de que disponen. La comunicación entre ordenadores de estos dos tipos se lleva a cabo utilizando comandos especiales. Las respuestas del ordenador DB pueden ser tanto un grupo de registros (conjunto respuesta) o simplemente indicando si se ha podido realizar o no un determinado comando (una determinada operación de búsqueda o escritura).

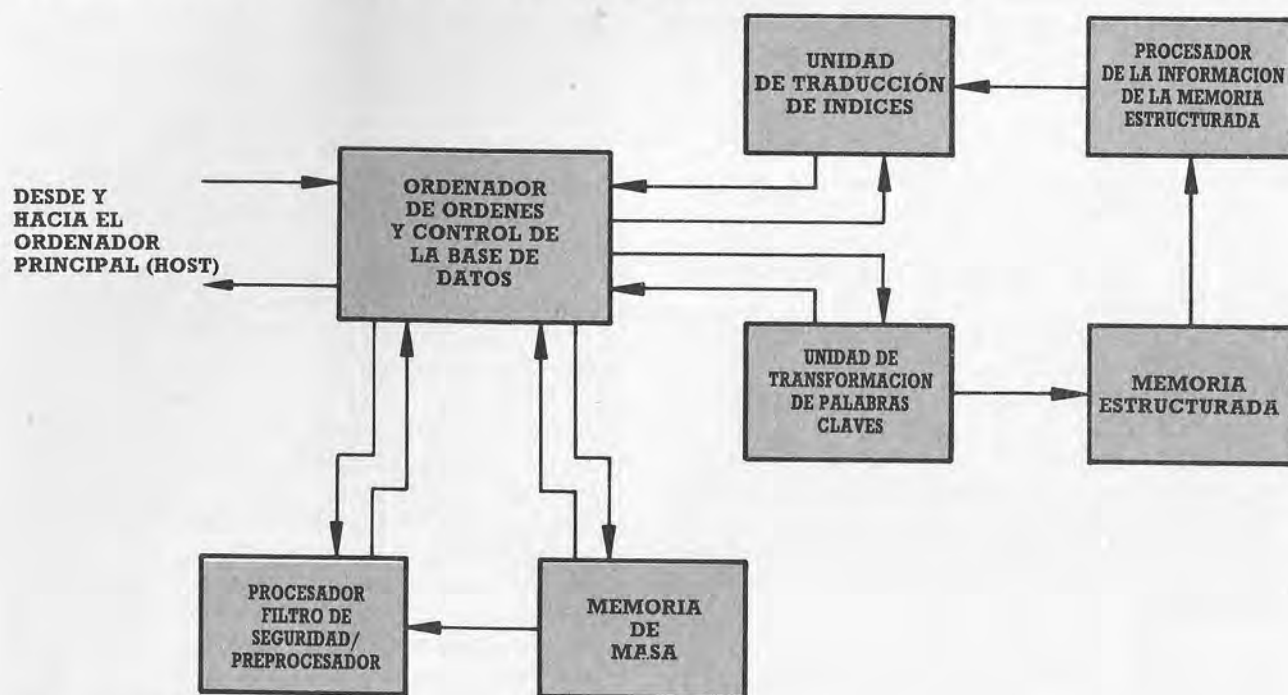
Para la ejecución de los comandos los ordenadores DB utilizan dos bucles de procesadores y memorias. El primero, el bucle de datos, está formado por el ordenador de órdenes y control de la base de datos, la memoria de masa, el procesador «post-acceso» y el procesador «filtro de seguridad», utilizados, respectivamente, para el almacenamiento y acceso a la base de datos, para una recuperación o proceso posterior de registros, y para alcanzar un mayor nivel de seguridad.

El otro bucle, llamado bucle estructural, consiste en el ordenador de órdenes y control de la base de datos, la unidad de transformación de palabras clave, la memoria estructurada, el procesador de información de la me-

moria estructurada y la unidad de traducción de índices, que se utilizan, respectivamente, para limitar el espacio de búsqueda de la memoria de masa, para localizar aquellos registros cuyo acceso se haya autorizado (control A), así como para agrupar (*clustering*) los registros recibidos para su inserción en la base de datos.

Las tecnologías utilizadas en este tipo de máquinas son muy avanzadas. Una de ellas, la memoria de masa on-line, utiliza discos y cabezas con movimiento (una técnica muy avanzada en dispositivos de almacenamiento on-line de muy alta capacidad). Estos discos, adaptados a la máquina, pueden soportar lecturas y escrituras en paralelo, sobre el cilindro completo, en cada revolución, frente a los sistemas tradicionales que van leyendo o escribiendo pista por pista. Este tipo de lectura agiliza considerablemente el acceso a la información. Los distintos canales de flujos de datos pueden ser dirigidos según el contenido (*content addressed*) simultáneamente por un conjunto adecuado de procesadores de información de pistas, en cada revolución. Estos equipos se encuentran físicamente en el controlador del disco, nunca en las propias cabezas de lectura/escritura de los discos. Es útil que se limite el acceso a unos cuantos cilindros, ya que las transacciones de un solo usuario rara vez se refieren a datos cuyo tamaño se mida en megabytes. Mientras los datos no se dispersen físicamente, se puede evitar la extracción y lectura de gran número de cilindros. La dispersión física de los datos relacionados es un fenómeno a evitar, y se logra mediante un mecanismo de agrupamiento de los datos (*clustering*) en el ordenador de órdenes y control de la base de datos, que utiliza información de los creadores de la base de datos, obtenida del ordenador principal (*host*).

Resulta evidente la utilidad de que la máquina disponga de información sobre la estructura de la base de datos que maneja. Si no dispusiera de dicha información, uno de los inconvenientes sería que por cada operación se utilizarían todos los cilindros, sin considerar si existe algún tipo de agrupamiento (*clustering*) o no. Otro problema que debe considerarse (y que no parece importante a primera vista) es el tema de la seguridad en el acceso a la base de datos. Por muchas razones que no vamos a exponer, el usuario necesita autorización para acceder a la base de datos. Los sistemas de comprobación y seguridad suponen aproximadamente el 10 % del tamaño de la base de



Esquema de organización de un sistema especializado en la gestión de bases de datos.

datos, y dado el volumen de éstas, es considerable. Como la velocidad de acceso a la información es un factor esencial, y el número de posibles usuarios puede ser grande, es evidente la importancia de las técnicas especializadas (basadas en el conocimiento de estructuras específicas de las bases de datos).

Dispositivos comunes en estas máquinas son memorias RAM dinámicas, y dispositivos de carga acoplada (*charge-coupled*). Otros componentes usuales son multiprocesadores, procesadores de flujo de datos dirigidos (*pipeline*), utilizados por el procesador de órdenes y control de la base de datos para proceso concurrente, mejorando las características generales del equipo. También debe existir un control y sincronización de todos los componentes del ordenador DB, para que funcionen concurrentemente sobre uno o más comandos. Estos son enviados al procesador de órdenes y control de la base de datos, desde el principal. El procesador de órdenes y control de la base de datos recibe los comandos del ordenador principal, y envía grupos de registros y otros mensajes de diagnóstico. El procesador de órdenes y control de la base de datos tiene muchas otras funciones, como, por ejemplo, el agrupamiento de registros durante la inserción, y el preproceso de las especificaciones de seguridad, tanto a nivel de campo (en el lazo de proceso de datos) como

a nivel de registro (en el bucle estructural). Además, coordina las tareas de comprobación de los accesos, indicando al procesador filtro de seguridad que debe volver a contrastar posteriormente las respuestas para el nivel de campo, control del procesador posacceso, para unir o no las respuestas, y otras muchas tareas.

Consideremos, por otro lado, la propia base de datos almacenada en la memoria de masa. Está formada por registros. Los registros, a su vez, constan de los siguientes elementos: los datos propiamente dichos, un conjunto de atributos (pares de valores llamados palabras clave) y un número representativo del grupo de registros (al que pertenece el que estamos considerando) y que satisfacen las mismas especificaciones de seguridad (por lo que en ocasiones se llaman átomos de seguridad). Estos átomos constituyen una partición lógica de la base de datos, ya que disponen de la misma protección con respecto a un usuario concreto. En realidad, la noción de grupo y la noción de átomo son una misma cosa, al menos en nuestra pequeña discusión. La memoria estructurada es el dispositivo que mantiene los índices de búsqueda, ya que la base de datos se encuentra en varios cilindros y sólo se puede inspeccionar un cilindro en cada momento. Por ejemplo, si tenemos una palabra clave, la entrada a la memoria estruc-

turada consistirá en una lista de términos del índice, en la que aparecen en el orden establecido la palabra clave, el cilindro y el átomo donde se encuentran los registros con la palabra clave de que hablamos.

Una vez dadas las conjunciones booleanas de los predicados que forman las palabras clave (también llamadas conjunciones de consulta) formando parte de un comando de entrada, el procesador de órdenes y control de la base de datos va considerando cada conjunción de consulta sucesivamente. La unidad de traducción de índices utiliza el atributo del predicado y el nombre del fichero para determinar (para cada predicado de la conjunción) la zona aproximada (en la estructura de la memoria) donde se encuentran aquellas palabras clave que satisfacen el predicado. La memoria estructurada recupera el conjunto de datos formado por los indicadores para aquellas palabras clave que satisfacen un predicado en una conjunción de consulta y a continuación los transmite al procesador de información de la memoria estructurada. Después, este dispositivo procesa la intersección de estos grupos de datos correspondientes a todos los indicadores. El número de grupos de datos es igual al número de predicados en la conjunción de consulta. Después de dicha intersección, el conjunto resultante de indicadores es filtrado por el procesador de órdenes y control de la base de datos, borrando todos aquellos indicadores cuyos números de átomo no están disponibles (autorizados) para el usuario. (En este caso, el usuario es aquella persona que realiza la consulta). A continuación, en la unidad de traducción de índices traduce estos indicadores en números correspondientes a cilindros, que, juntamente con la conjunción de consulta completa, se envían a la memoria de masa, para buscar su contenido. Más aún, la salida de la memoria de masa puede volver a ser procesada y comprobada de nuevo por el procesador filtro de seguridad, antes de volver a ser enviada al ordenador principal de que se trate.

Dentro de los comandos de entrada podemos distinguir dos tipos distintos: los comandos de acceso y los preparatorios. Los co-

mandos de acceso son aquellos que incluyen algún acceso a la memoria de masa, y los preparatorios son los que transfieren la información sobre la base de datos, como, por ejemplo, los nombres y atributos de los ficheros que se van a crear, sus características, espacio disponible y las especificaciones de seguridad. Los comandos de acceso se ejecutan por procesos pipeline (conducción, hemos hablado repetidamente de este tipo de procesos) por los componentes del sistema completo de hardware de manejo de la base de datos. Los comandos de acceso se procesan en primer lugar (según sus indicadores), por el bucle estructural de procesadores y memoria para determinar el número del cilindro adecuado. El procesador de órdenes y control de la base de datos coordina la operación de los restantes componentes, llevando la cuenta del estado de los comandos que se están ejecutando en ese momento. La información de los comandos preparatorios se almacena organizada en una memoria de acceso aleatorio de procesador de órdenes y control de la base de datos. Esta información es referenciada durante la ejecución de los comandos de acceso en numerosas ocasiones.

Por otro lado, el procesador de órdenes y control de la base de datos, DBCCP, agrupa los registros que se van a incluir en la base de datos, considerando sus atributos de agrupamiento (*clustering*).

Podríamos estudiar muchas otras características de este tipo específico de ordenadores, que evidentemente no caben en estas líneas. Los posibles sistemas de manejo de las «colas de acceso a la información» ofrecen un estudio muy interesante, y son una de las características especiales de estos equipos. Otro punto de importancia capital son las comprobaciones analíticas y valoración de características de los distintos sistemas, para los que se pueden seguir distintas técnicas. Un estudio de tiempos también resulta fundamental, y en muchos casos los métodos experimentales (como la simulación) pueden resolver numerosas incógnitas, tanto en el diseño como en el estudio de las características y operatividad de una base de datos.

Backend, procesador. Procesador diseñado para una tarea específica. Puede estar especializado, por ejemplo, en la gestión de una base de datos, o una unidad aritmética y lógica. Se suelen designar también con los nombres castellanos «ordenador dedicado», u ordenador de una sola tarea.

Benchmark. Conjunto de pruebas a que se somete un determinado sistema o equipo para evaluar consiguientemente su rendimiento y medir sus características (parámetros representativos del sistema). En dicho sistema a evaluar se suelen incluir máquinas y programas, aunque en ocasiones también se definen *benchmarks* específicos para cada uno de estos elementos. En este tipo de pruebas, el sistema se encuentra sometido a una carga de trabajo conocida, y se van midiendo rendimientos. Los *benchmarks* se realizan siempre para efectuar pruebas comparativas con otros equipos y software, a los que se ha sometido también al *benchmark*, y cuyos rendimientos son conocidos.

Cluster. Término inglés que literalmente significa ramillete, racimo. En informática, se refiere a algún tipo de agrupamiento, que puede ser de cintas magnéticas, discos y otros dispositivos o elementos, y que de alguna forma deben ser considerados como una unidad o tratados conjuntamente. En muchas ocasiones, el agrupamiento está controlado por un maestro. A veces, existen dos caminos de acceso a un único agrupamiento o cluster de dispositivos (cada uno de estos caminos con su correspondiente unidad de control), para agilizar el acceso al cluster y obtener mayor seguridad frente a la posibilidad de fallo de uno de los dos controladores.

Comandos de acceso. En los ordenadores de manejo de bases de datos se designa con este nombre a las instrucciones que requieren durante su ejecución uno o más accesos a las memorias de masa del sistema.

Comandos preparatorios. En los ordenadores de manejo de bases de datos se designa con este nombre a las instrucciones y procesos que se realizan para la preparación de los índices de acceso a la memoria de masa, localización de los datos dentro de dicha memoria, etc.

Content-addressed, memoria. Memoria que puede ser direccionada por el contenido además de por las direcciones. También se suelen designar como memorias asociativas.

DBC. Data Base Computer-Procesador de bases de datos. Procesador dedicado a la gestión de bases de datos, por lo que dispone de una estructura especialmente apta para el manejo de las direcciones de los datos y para realizar accesos en paralelo, agrupamiento de datos para su lectura y/o escritura, etc.

Dedicado, procesador dedicado. Procesador especializado en alguna tarea, en contraposición a ordenador de propósito general. En el ordenador dedicado, prima la eficacia y la alta rentabilidad que se puede obtener del procesador para realizar la tarea para la que ha sido diseñado, frente a la versatilidad y flexibilidad de los ordenadores de propósito general.

Frontend, procesador. Procesador de comunicaciones. Procesador de entrada/salida que se utiliza para realizar el formato de los datos de entrada salida, gestionar la red



de comunicaciones, establecer los protocolos de diálogo con los terminales, dirigir los mensajes, etc., liberando de esta tarea al ordenador principal.

Host, procesador. Procesador principal. Se utiliza este término para referirse al ordenador que dirige las comunicaciones, en un entorno complejo de terminales y periféricos. En un grupo de ordenadores con arquitectura distribuida se considera "host" al que gobierna el funcionamiento global de la configuración, distribuye tareas a los otros ordenadores y coordina los flujos de datos

entre ellos. Estos otros ordenadores, y en general, los dispositivos dependientes de un host cualquiera, suelen ser llamados «slave» (esclavos).

Memoria de masas. Sistema de almacenamiento masivo auxiliar en línea, donde pueden almacenarse grandes cantidades de datos. Se suele reservar este término para los dispositivos que permiten almacenamiento en grandes volúmenes de información, y no tanto para los «clusters» o agrupamientos de dispositivos menores.

Máscara. Esquema de campos prefijados utilizado para representar o procesar datos. Cuando aparecen en pantalla favorecen la introducción adecuada de los datos por parte del operador. También puede referirse a un conjunto de bits que quedan indicados para que sobre ellos se ejecute la operación adecuada, ocultando los restantes.

Mask. (Ver **Máscara**)

Mass memory. (Ver **Memoria de masa**).

Master. (Ver **Maestro**.)

Master/slave. (Ver **Maestro/esclavo**.)

Match. Término inglés que significa concordar. En informática se utiliza en comparaciones de dos o más elementos o datos.

Matemática, comprobación. Comprobación programada que utiliza relaciones matemáticas. También se utiliza el término comprobación aritmética.

Matemático, modelo. Representación matemática de un proceso, dispositivo o concepto.

Matricial, almacenamiento. Almacenamiento organizado de tal forma que el acceso a cualquier dirección requiere utilizar dos o más coordenadas o subíndices. La dirección de un dato viene dada por el número de la fila y columna de la matriz, en cuya intersección se encuentra dicho dato.

Matriz. Conjunto de valores dispuestos en forma de tabla de subíndices que representa los valores de una tabla de varias dimensiones. Cuando las matrices sólo tienen un único índice, se llaman vectores. Las matrices se tratan matemáticamente cumpliendo las normas del álgebra de matrices. Tam-

bién puede referirse a una red local en forma matricial, con entradas y salidas diversas, y los elementos lógicos conectados en las intersecciones.

Matriz de puntos. Técnica por la cual se pueden escribir caracteres alfanuméricos y gráficos utilizando puntos de una matriz rectangular. El sistema de matriz de puntos se utiliza en algunas impresoras y para la representación de caracteres en las pantallas. Valores usuales de las matrices de puntos utilizadas en las impresoras son 7×9 , 9×12 , etcétera.

Mayor o igual que. Operador de relación entre dos valores. Para designarlo se utilizan los símbolos $\geq$. La igualdad entre las dos cantidades es también válida.

Mayor que. Operador de relación que suele designarse con el símbolo $<$.

Media. Función matemática que obtiene el valor medio entre varios. Para ello, suma todos los valores y divide el total entre el número de valores tomados.

Medio. Material o configuración en el que se registran los datos. Puede referirse a cintas magnéticas, tambores, etc.

Mega. Prefijo multiplicador que equivale a un millón. 7 Megas = 7 millones.

Megabit. Un millón de bits.

Megabyte. Un millón de bytes.

Memoria. Sistemas de almacenamiento. Existen muchos tiempos de memorias. Véase a continuación el tipo que se desee.

Memoria caché. Memoria muy rápida que se utiliza como memoria intermedia (*buffer*) entre la memoria central y la unidad de con-

trol del ordenador. Se utiliza en ordenadores de dimensiones grandes o medios, y su misión es agilizar al máximo los accesos a la memoria central, reduciendo al mínimo los tiempos medios de acceso a dicha memoria central del sistema.

Memoria de acceso directo. Memoria cuyos elementos son todos ellos accesibles directamente. A este tipo pertenecen las memorias centrales (RAM y ROM), las memorias cache, etc.

Memoria de burbujas. Memoria de almacenamiento masivo en la cual los soportes magnéticos no están en movimiento. Los elementos constitutivos son cilindros muy pequeños, llamados burbujas, cada una de las cuales lleva la información de un bit. La información se graba en las burbujas de forma secuencial, mediante una cabeza de escritura. Una vez realizada la operación, las burbujas se desplazan mediante un campo magnético, hasta su soporte, llamado silo. Para la lectura, el proceso es inverso, sacando las burbujas de los silos y haciéndolas pasar por una cabeza de lectura.

El tamaño de las burbujas es aproximadamente de una diezmillonésima de metro. Sus ventajas principales son que los datos se pueden retener aunque falte la alimentación, y los beneficios derivados de no necesitar partes mecánicas en movimiento.

Memoria central. Una de las partes del ordenador donde se almacenan los programas que se van a ejecutar y los datos que dichos programas van a utilizar en su ejecución. Suele denominarse simplemente RAM (Random Access Memory-Memoria de Acceso Aleatorio), por disponer de esta característica. La memoria está formada por pequeñas celdillas, cada una de ellas asociada con una dirección y donde se pueden almacenar un número fijo de bits. El acceso, como hemos indicado antes, es directo a cada una de las celdas.

Memoria intermedia. Llamada generalmente con el término inglés *buffer*. Es una zona de la memoria que se utiliza como tampón durante las operaciones de entrada/salida. En ella se almacenan temporalmente aquellos datos que se van a transferir. Estas memorias se emplean cuando los dos dispositivos que van a transmitir/recibir los datos tienen velocidades de transmisión diferentes.

Memoria de masa. Unidad periférica adecuada para almacenar grandes cantidades de datos. Pueden ser, por ejemplo, unidades de disco, tambores magnéticos, etc. Su tiempo de acceso es algo superior al de acceso a la memoria central, pero su utilización tiene otras ventajas. (Básicamente, el gran volumen de datos que se pueden almacenar, transportabilidad, etc.).

Memoria de núcleos. Memoria central de tipo magnético realizada con pequeños toros o núcleos de ferrita. Para almacenar los dos posibles valores de un bit, 0 ó 1, se magnetiza una de las dos caras del toro. Éste tipo de memoria es muy antiguo y actualmente han sido sustituidas por otros modelos de memorias electrónicas. Su ventaja fundamental es que al apagarse la alimentación los datos almacenados en ella no se pierden. Sin embargo, son más voluminosas, el acceso a la información es más lento y su precio tampoco puede competir con los modernos tipos de memorias electrónicas de nuestros ordenadores actuales.

Memoria de trabajo. Zona de la memoria en la que se colocan temporalmente los datos. También se utiliza para designarla el término memoria intermedia o *buffer*.

Memoria secundaria. Este término se utiliza en contraposición a las memorias centrales. Se refiere a las memorias periféricas (discos rígidos, disquettes, cassettes, etc.). a las memorias centrales se les denomina primarias, para distinguirlas de este tipo de memorias.

Memoria virtual. Espacio de trabajo definido en un sistema para el almacenamiento de datos y/o programas que no se corresponden con la memoria real (física) central existente, sino que la rebasa ampliamente. La información que se considera almacenada en esta memoria, de hecho, estará en parte en la memoria central RAM del ordenador, y en parte en un dispositivo externo de almacenamiento de acceso rápido, de modo que cuando se necesite un dato, el sistema de gestión de la memoria virtual «sabe» si está en la parte («página») o partes almacenadas en la memoria directa (y accede a ella), o no, en cuyo caso provoca un comando de lectura de la página desde el dispositivo externo, para su posterior acceso inmediato. Con la memoria virtual se consigue disponer de un espacio o área de memoria para

trabajo mucho mayor que la memoria real disponible. Sin embargo, el overhead (carga adicional de trabajo) que el proceso de lectura y escritura comporta, puede no compensar, en ocasiones, las otras ventajas que la memoria virtual reporta.

Memoria volátil. Memorias cuyo contenido no perdura por tiempo indeterminado. La mayoría pierde la información cuando se desconecta la alimentación, aunque existen otros tipos que necesitan reescribir periódicamente su contenido.

